24/10/2024

## Projeto Finanças – aplicação Python

## Chave SSH:

1. Verificar a existência de chave: ls -al ~/.ssh
2. Gerar uma nova chage: ssh-keygen -t ed25519 -C [accolombini@gmail.com](mailto:accolombini@gmail.com)
3. Adicionar a chave SSH ao agente SSH: eval "$(ssh-agent -s)"
4. ssh-add ~/.ssh/id\_ed25519
5. Adicionar a chave SSH ao GitHub – primeiro copie a chave gerada: cat ~/.ssh/id\_ed25519.pub
6. Copie o conteúdo exibido, no GitHub vá em: Acesse o GitHub, vá para **Settings > SSH and GPG keys > New SSH key**, dê um nome para a chave e cole o conteúdo no campo.
7. Teste a conexão com o GitHub: ssh -T [git@github.com](mailto:git@github.com)
8. Tudo estando correto, teremos uma mensagem com: Se tudo estiver correto, você deverá ver uma mensagem de boas-vindas.

Projeto ARFinancas - retomado no algoritmo **t20**\_**final**\_**testes**.**py**

* Projeto sendo remodelado para analisar portfólio todo
* Muitas alterações e dificuldades para encontrar um ponto de conversão
* Algumas sugestões interessantes foram passadas, no momento seguimos como o que temos.

## Reunião semanal com Petro

1. Reunião Petro – 11:00
2. Reunião TCC Letícia agendada para o dia 25/10 – 11:30
3. Propor reunião com Fabiano – revisão do artigo – sugestão 16:00 – Classroom
   1. Preparar material para revisão do artigo Fabiano - feito.

**Reunião com a Petro:**

* Marcos pede que façamos uma leitura do material que ele enviou. Nota que não estamos falando da fase 3.
* Controle de acesso ok
* A ferramenta definida foi o SNS em substituição ao RabbitMQ
* Os alunos Bruno, Davi e Júlia foram atualizados no contexto Soraia de gestão
* Pipeline e SONAR em fase de teste pelo time UFF – (entre a semana que vem e a próxima)
* Já foram autorizadas as compras das máquinas – isto deve demorar por volta de 30 dias.

## Projeto ArFinancas

**Módulos de atuação:**

1. T20\_md1\_coleta\_processamento.py
2. T20\_md2\_visualizacao\_daschboard.py
3. T20\_md3\_prev\_curto\_prazo.py
4. T20\_md4\_prev\_longo\_prazo.py
5. T20\_md5\_esemble\_modelos.py

**Estratégia abordada para estes módulos:**

**Proposta de Divisão do Módulo 1**

**Parte 1: Coleta e Processamento dos Dados dos Ativos**

* Responsável por baixar e processar os dados dos ativos (ações e índice Bovespa), aplicando as transformações e extraindo as features (médias móveis, volatilidade, etc.).
* Os dados seriam então salvos em um CSV ou retornados para uso posterior.

**Parte 2: Coleta e Integração dos Indicadores Econômicos**

* Responsável por buscar os indicadores econômicos da API do FRED ou qualquer outra fonte relevante.
* Integração opcional desses indicadores aos dados processados anteriormente.

**Divisão proposta para o Módulo 3: Modelos de Previsão (Curto e Longo Prazo)**

Parte 1: Previsão de Curto Prazo (curto\_prazo.py)

* Conteúdo: Modelos de previsão para o curto prazo (ex.: LSTM, XGBoost, LightGBM)
* Foco: Capturar variações rápidas em períodos curtos (30 dias).
* Modelos:
* LSTM para padrões temporais sequenciais.
* XGBoost / LightGBM para captar variações imediatas nos dados dos ativos e indicadores econômicos.
* Validação: Validar as previsões no conjunto de validação de curto prazo (30 dias).

**Parte 2: Previsão de Longo Prazo (longo\_prazo.py)**

* Conteúdo: Modelos de previsão para o longo prazo (ex.: Prophet, ARIMA, LSTM multivariado)
* Foco: Capturar tendências e sazonalidades de longo prazo (2 a 5 anos).
* Modelos:
* Prophet para tendências de longo prazo e sazonalidades.
* ARIMA para séries temporais estatísticas.
* LSTM multivariado para capturar padrões complexos entre diferentes ativos e indicadores econômicos.
* Validação: Implementar intervalos de confiança para previsões de longo prazo e avaliar o desempenho.

**Parte 3: Ensemble de Modelos (ensemble.py)**

* Conteúdo: Combinação dos modelos de curto e longo prazo.
* Foco: Criar um ensemble para aproveitar as forças de cada abordagem.
* Métodos:
* Média ponderada: Simples combinação de previsões ponderadas.
* Stacking: Combinação de modelos com um meta-modelo, ajustando o peso de cada previsão.
* Saídas: Previsões finais com intervalos de confiança ajustados e interpretação da contribuição de cada modelo.

## Importante análise e desempenho dos ativos frente ao BOVESPA

O Ibovespa, como um índice ponderado de ações, reflete o comportamento das principais ações da bolsa brasileira. Isso significa que, em geral, ele tende a capturar movimentos de alta e baixa do mercado de maneira mais balanceada.

Se os ativos individuais têm dificuldade em superar o índice, pode ser um sinal de que as escolhas desses ativos não estão superando o desempenho geral do mercado no período analisado.

**Volatilidade dos Ativos:**

Observando o gráfico, é possível ver que alguns ativos apresentam maior volatilidade em relação ao Ibovespa. Isso pode significar que esses ativos têm um risco maior e, apesar de poderem gerar retornos maiores em alguns momentos, não conseguem manter a consistência no longo prazo.

Ativos como VALE3 e PETR4 tendem a se alinhar ao Ibovespa, mas outros, como AMZO34, parecem mais voláteis.

**Períodos de Sobressaliência:**

Você pode investigar os momentos específicos em que certos ativos superam o índice. Isso pode ocorrer em contextos específicos (como crises ou períodos de alta demanda por commodities, que afetam ações como VALE e PETROBRAS).

O que pode ser feito?

**Revisitar a Seleção de Ativos:**

Avaliar se os ativos escolhidos estão realmente alinhados com a estratégia de investimento. Ativos que não conseguem superar o Ibovespa consistentemente podem ser substituídos ou complementados por outros ativos com potencial de desempenho superior.

**Avaliar Estratégias de Investimento:**

Considere estratégias que utilizem o Ibovespa como benchmark. Se o objetivo é superá-lo, pode ser interessante buscar ativos com menor correlação ao índice ou explorar setores menos representados no índice.

**Fazer um Estudo Mais Detalhado:**

Podemos calcular métricas adicionais, como índice de Sharpe e alfa, para entender se o desempenho ajustado ao risco dos ativos justifica mantê-los em um portfólio. Também podemos investigar em quais contextos (internacionais ou domésticos) esses ativos performam melhor ou pior em relação ao Ibovespa.

## Refatorando para atender a complexidade das séries temporais financeiras

**Análise dos Algoritmos e Recomendações**

LSTM (Long Short-Term Memory):

Embora o LSTM seja adequado para capturar dependências temporais, ele não necessariamente captura bem as correlações complexas entre ativos financeiros, especialmente em uma configuração não multivariada.

Sugiro considerar um LSTM multivariado que inclua múltiplas séries (ativos) em uma mesma entrada de rede, o que poderia capturar interdependências entre os ativos e indicadores econômicos.

Alternativamente, podemos utilizar um modelo seq2seq LSTM, que permite um ajuste mais fino para previsões em sequência e pode melhorar a precisão para séries temporais financeiras.

Modelos Baseados em Árvores (XGBoost e LightGBM):

Modelos baseados em árvores, como XGBoost e LightGBM, têm se mostrado robustos em captar relações complexas sem necessariamente depender de estruturas temporais, o que pode ser uma vantagem.

Esses modelos podem ser usados em conjunto com variáveis explicativas adicionais (ex., volatilidade, correlações entre ativos), o que poderia enriquecer as previsões e ajudar a lidar com a complexidade dos dados financeiros.

Modelos de Séries Temporais Financeiras Específicos:

Modelos como GARCH ou EGARCH são frequentemente usados para prever volatilidade e são adequados para dados financeiros voláteis. Eles poderiam ser incluídos como uma camada adicional para entender melhor a volatilidade e a variabilidade ao longo do tempo.

Prophet ou ARIMA com uma abordagem multivariada também podem ser adequados para capturar tendências e sazonalidades gerais.

Ensemble Estratégico:

Podemos ajustar o ensemble para ponderar mais fortemente os modelos que capturam melhor os diferentes aspectos da série temporal. Uma sugestão é considerar um stacked ensemble, em que um meta-modelo aprenda quais modelos têm melhor desempenho para certos padrões de dados.

Próximos Passos com Refatoração do Código

Implementação de um LSTM Multivariado:

Ajustar o código para usar múltiplos ativos na entrada do modelo LSTM e incluir indicadores econômicos como entradas adicionais.

Incorporação de Modelos Específicos para Financeiros:

Avaliar o uso de GARCH para previsão de volatilidade, combinando suas saídas como variáveis de entrada para os modelos de previsão principais (LSTM, XGBoost, LightGBM).

Refatoração do Ensemble:

Implementar um ensemble que ajuste os pesos dinamicamente ou utilize um modelo meta para aprender a combinação ideal dos modelos.

**Visualização e Validação:**

Revisar o código de visualização para garantir que estamos exibindo as previsões de forma clara e ajustada ao horizonte de curto prazo.

## Abordagem dividir em módulos menores (algoritmos de ML)

**Estrutura Sugerida de Arquivos e Módulos**

1. **Scripts de Treinamento Individual para Cada Modelo**:
   * **LSTM Multivariado** (train\_lstm\_multivariado.py): LSTM ajustado para múltiplas variáveis.
   * **Seq2Seq LSTM** (train\_seq2seq\_lstm.py): Para capturar ajustes mais finos.
   * **Modelos Baseados em Árvores**:
     + **XGBoost** (train\_xgboost.py): Modelo para capturar relações de curto prazo.
     + **LightGBM** (train\_lightgbm.py): Modelo alternativo ao XGBoost para comparar performances.
   * **Modelos de Volatilidade (GARCH/EGARCH)**:
     + **GARCH ou EGARCH** (train\_garch.py): Para prever a volatilidade de retornos e fornecer insights auxiliares.
   * **Modelos de Séries Temporais**:
     + **ARIMA** (train\_arima.py): Modelo estatístico para prever tendências.
     + **Prophet** (train\_prophet.py): Modelo sazonal e de tendência, bom para séries temporais com padrões bem definidos.
2. **Script para Avaliação de Resultados Individuais** (evaluate\_models.py):
   * Este script seria responsável por carregar as previsões de cada modelo e comparar as métricas de desempenho, como RMSE e MAPE, com os dados reais.
   * Após cada execução, registrar os resultados para fácil comparação entre modelos.
3. **Estratégia de Integração**:
   * **Ensemble (apenas após a análise individual)** (ensemble\_models.py): Integrar os melhores modelos após avaliação individual. Podemos, então, experimentar métodos de ensemble (como stacking ou média ponderada) para combinar as previsões dos modelos mais promissores.

**Fluxo de Trabalho Proposto**

1. **Implementar e Avaliar Cada Modelo Separadamente**:
   * Desenvolver cada script de treinamento individual, testando e ajustando cada modelo isoladamente.
   * Avaliar e registrar as métricas para cada um e identificar os melhores modelos para o tipo de dados.
2. **Comparação de Desempenho**:
   * Usar o script de avaliação para comparar o desempenho dos modelos. Esta etapa garantirá que cada modelo seja avaliado em pé de igualdade.
3. **Seleção e Ajuste do Ensemble**:
   * Com base nos resultados dos testes individuais, escolher os modelos mais robustos e implementar o ensemble.
   * Ajustar o ensemble com base na análise dos resultados.
4. **Documentação e Análise Final**:
   * Documentar o desempenho de cada modelo e os motivos para incluir ou não no ensemble final.
   * Finalizar com visualizações que mostrem o desempenho preditivo dos modelos ao longo do tempo.

## Refinando o processo para médio prazo (1 a 2 anos)

Com um histórico de 10 anos e um horizonte de previsão de pelo menos dois anos, a modelagem se torna mais complexa, pois o objetivo passa a ser a previsão de médio a longo prazo, onde características como tendências, sazonalidades e choques externos desempenham um papel significativo. Para abordar isso de forma estratégica e selecionar os algoritmos mais adequados, aqui está uma análise aprofundada dos ajustes que podemos fazer:

**Estrutura da Estratégia de Modelagem**

1. **Dividir os Objetivos: Previsão de Curto e Longo Prazo**
   * **Curto Prazo** (próximos 30-90 dias): Focar em capturar as oscilações e volatilidade, essencialmente usando modelos como LSTM multivariado e modelos baseados em árvores (XGBoost, LightGBM) com ajuste fino nas entradas e sequências temporais mais curtas.
   * **Médio e Longo Prazo** (1-2 anos): Modelos que capturam tendências e sazonalidades de longo prazo, como ARIMA e Prophet, são essenciais. Além disso, algoritmos que podem processar múltiplas séries temporalmente correlacionadas, como LSTM seq2seq e modelos GARCH para volatilidade, também são importantes. Por fim, o ensemble final pode combinar previsões para obter uma robustez maior.

**Algoritmos Recomendados e Parâmetros-Chave**

1. **LSTM Multivariado (seq2seq) para Curto e Médio Prazo**
   * **Configuração:** Aumentar o sequence\_length para 120-180 dias, permitindo ao modelo capturar sazonalidades anuais e padrões mais amplos.
   * **Modelo seq2seq:** Em vez de prever uma única sequência, utilizaremos uma estrutura de seq2seq (sequência para sequência), onde o modelo é ajustado para aprender a sequência de entrada e, em seguida, gerar previsões iterativas.
   * **Número de Unidades LSTM:** Usar uma rede profunda com 200-300 unidades nas camadas principais LSTM, além de uma camada densa final.
   * **Batch Size e Epochs:** batch\_size=64 e epochs=200 para garantir que o modelo tenha capacidade de generalização suficiente.
   * **Saída Multihorizon:** Configurar o modelo para gerar previsões de 60 a 90 dias por iteração, acumulando assim a previsão para dois anos.
2. **Modelos de Séries Temporais como ARIMA e Prophet**
   * **Prophet:** Ideal para capturar tendências de longo prazo e sazonalidades anuais, com ajuste de eventos sazonais específicos e feriados.
   * **ARIMA com parâmetros SARIMA (sazonal):** Para séries com padrões claros de sazonalidade e tendência. Com uma ordem sazonal adequada, é possível modelar as flutuações sazonais e tendências de longo prazo.
   * **Configuração:** Treinar Prophet e ARIMA em horizontes de um a dois anos, ajustando hiperparâmetros sazonais para obter previsões confiáveis para o horizonte de 1-2 anos.
3. **GARCH para Captura de Volatilidade e Incorporação de Incerteza**
   * **GARCH/EGARCH** são particularmente úteis para séries financeiras, capturando bem a volatilidade. Esses modelos podem ser ajustados para gerar previsões da volatilidade a cada dia.
   * **Integração com Outros Modelos:** As previsões de volatilidade podem ser utilizadas como uma entrada para os modelos LSTM ou como parte do ensemble, ponderando as previsões de acordo com o nível de incerteza.
4. **Modelos Baseados em Árvores (XGBoost e LightGBM) com Features Engenheiradas**
   * **Configuração:** Esses modelos podem ser treinados com variáveis derivadas adicionais, como volatilidade prevista, médias móveis de 30, 60 e 90 dias, e correlações entre os ativos.
   * **Interpretação:** Modelos de árvores são mais interpretáveis, permitindo que observemos quais variáveis têm maior impacto na previsão.
   * **Parâmetros:** max\_depth=8, learning\_rate=0.01, e n\_estimators=500 para XGBoost e LightGBM. Eles capturam melhor a não linearidade nas relações entre os ativos.
5. **Ensemble Ponderado para Combinação de Modelos**
   * **Configuração de Ensemble:** Utilizar um ensemble ponderado, onde modelos como Prophet e ARIMA têm maior peso para previsões de longo prazo, enquanto o LSTM multivariado e GARCH têm pesos mais altos para previsões de curto prazo.
   * **Stacking:** Em uma etapa futura, podemos experimentar o uso de um modelo meta para aprendizado das combinações ideais. Stacking ajuda a ajustar automaticamente os pesos e melhorar a robustez geral.

**Estrutura do Código e Implementação por Etapas**

1. **Dividir o código em scripts separados por algoritmo**:
   * **Script para LSTM Multivariado e Seq2Seq.**
   * **Script para Prophet e ARIMA.**
   * **Script para GARCH e análise de volatilidade.**
   * **Script para Modelos de Árvores (XGBoost e LightGBM).**
   * **Script final para Ensemble.**
2. **Avaliação e Seleção de Modelos**
   * Executar cada modelo de forma independente para entender qual deles tem o melhor desempenho em previsões de curto e longo prazo.
   * Documentar os RMSE e outras métricas de desempenho, observando os algoritmos que oferecem a melhor precisão ao longo dos períodos.
   * Baseado nas métricas e na análise visual, selecionaremos os modelos que melhor capturam as variações e tendências, eliminando aqueles que têm um desempenho significativamente inferior.

**Próximos Passos**

1. **Implementar o LSTM Multivariado com ajustes para previsão seq2seq e configurá-lo para as previsões de longo prazo.**
2. **Implementar Prophet e ARIMA para previsões de 1-2 anos, configurando seus parâmetros para capturar sazonalidades e tendências.**
3. **Incorporar GARCH para entender e modelar a volatilidade, gerando insights adicionais sobre o risco e incerteza nas previsões.**
4. **Executar XGBoost e LightGBM com features engenheiradas.**
5. **Montar o ensemble, priorizando os modelos mais robustos de acordo com os resultados obtidos.**

## Como treinar os modelos

Então, vamos ser bem claros, eu achei que você tinha guardado nossas conversas anteriores e pelo visto não fez isso. Quando falamos de 10 anos de coleta de dados estamos nos referindo ao período de 09/2014 a 10/2024, neste contexto nosso projeto vislumbra o seguinte 9/2014 a 9/2024 são dados para treinamento e testes do modelo. Os 30 últimos dias, isto é o mês 10/2024 deve ser usado de forma específica para validar o modelo. Quando tivermos todos os modelos ajustados para esse curto período de 30 dias, escolheremos o melhor trabalhamos o emsemble e usamos Monte Carlo para propor cenários de 2 a 5 anos no futuro, aí aplicamos nossos modelos nos cenários e tentamos encontrar qual o de melhor impacto e precisão.

### Proposta para atender ao escopo refinado:

**Estrutura do Projeto de Modelagem**

**1. Divisão de Dados e Período de Validação**

* **Treinamento e Teste:** Vamos usar o período de 09/2014 a 09/2024. Este intervalo será dividido internamente para treinar e testar o modelo. Utilizaremos métodos como validação cruzada em série temporal (TimeSeriesSplit) para otimizar o treinamento e ajustar hiperparâmetros, garantindo que o modelo generalize bem.
* **Validação Futura:** O mês de 10/2024 será reservado exclusivamente como conjunto de validação. Este será o primeiro passo para testar se o modelo é capaz de prever com precisão dados recentes e não vistos.

**2. Configuração dos Modelos**

* **Modelos Multivariados (LSTM, Seq2Seq LSTM)**: Ajustaremos hiperparâmetros e arquitetura com base nos 10 anos de dados históricos, incluindo séries multivariadas que englobem ativos diversos e indicadores econômicos, conforme discutido anteriormente.
* **Ajuste Fino de Hiperparâmetros**: Usaremos validação cruzada em série temporal para ajustar camadas, número de neurônios e outros parâmetros, visando um bom equilíbrio entre precisão e generalização.

**3. Validação e Escolha do Melhor Modelo**

* **Desempenho no Mês de Validação (10/2024)**: Testaremos cada modelo no período reservado de 10/2024 para avaliar sua precisão em um cenário próximo ao presente. Essa validação será crucial para selecionar o melhor modelo.
* **Comparação de Modelos e Escolha Final**: Escolheremos o modelo mais preciso para o mês de 10/2024 como o candidato ideal para compor o ensemble final.

**4. Ensemble e Cenários com Monte Carlo**

* **Ensemble**: Após a seleção do melhor modelo, integraremos os outros modelos para formar um ensemble que capture os diferentes aspectos das séries temporais (volatilidade, tendências e sazonalidades).
* **Simulações de Monte Carlo para Previsão de 2 a 5 Anos**: Usaremos o ensemble no conjunto de dados simulado por Monte Carlo para prever diferentes cenários econômicos. Cada cenário será analisado para identificar as previsões com maior impacto e precisão.

**Refatoração do Código para a Nova Estrutura**

Dado o que foi discutido, farei as seguintes alterações no código:

1. **Preparação de Dados**: Ajustarei a função prepare\_data\_seq2seq para garantir que os dados de 10/2024 sejam usados exclusivamente na validação. Dividirei o conjunto de dados conforme a estrutura discutida.
2. **Modelo Seq2Seq e LSTM Multivariado**: Farei ajustes na configuração e nos hiperparâmetros para aumentar a precisão e generalização.
3. **Inversão de Normalização e Visualização**: Garantirei que as previsões e os valores reais estejam corretamente invertidos para visualização e comparações precisas no período de validação.
4. **Plotagem e Análise**: Usaremos o Plotly para gráficos de previsão e comparação. Os gráficos serão focados em 10/2024 para validação e, posteriormente, na simulação de cenários futuros.

### Especificações dos dados:

1. **Baixe os dados históricos** dos ativos:
   * 'VALE3.SA' (Vale)
   * 'PETR4.SA' (Petrobras)
   * 'ITUB4.SA' (Itaú Unibanco)
   * 'PGCO34.SA' (Procter & Gamble - BDR)
   * 'AAPL34.SA' (Apple - BDR)
   * 'AMZO34.SA' (Amazon - BDR)
   * '^BVSP' (Ibovespa)
2. **Prepare os dados**:
   * Organize o histórico de 10 anos (de 09/2014 a 10/2024).
   * Divida o período de treino e teste, com os 30 dias finais de 10/2024 destinados para validação.
3. **Salve os dados preparados**:
   * Salve em TESTES/DADOS/train\_data.csv para facilitar o carregamento nos scripts de treinamento e validação.

### O roteiro para o script será:

1. **Baixar os Dados do Yahoo Finance**:
   * Usaremos a biblioteca yfinance para buscar dados dos ativos mencionados.
   * Os dados serão ajustados para o período de 09/2014 a 10/2024.
2. **Preparar os Dados**:
   * Organizaremos o dataset de modo que inclua:
     + Colunas de preços ajustados para cada ativo.
     + Transformação em um único DataFrame para fácil manipulação nos modelos.
   * Divisão automática entre treino (09/2014 a 09/2024) e validação (30 dias de 10/2024).
3. **Salvar em train\_data.csv**:
   * Salvaremos os dados formatados em TESTES/DADOS/train\_data.csv, prontos para serem usados diretamente nos scripts de modelagem.

### Indicadores macroeconômicos:

**Como os Indicadores Econômicos Podem Ajudar**

1. **Inflação e Taxa de Juros**:
   * Taxas de juros e inflação tendem a influenciar o fluxo de capital no mercado, afetando o custo de oportunidade e o apetite por risco.
   * Incorporá-las ao modelo pode ajudar a capturar as flutuações em resposta a mudanças políticas ou econômicas, proporcionando uma visão mais robusta do mercado.
2. **PIB**:
   * O crescimento do PIB está frequentemente relacionado à confiança do mercado, expansão econômica e retorno sobre os ativos.
   * Adicionar o PIB pode ajudar o modelo a entender a estabilidade econômica de longo prazo e suas flutuações sazonais.

**Estratégia de Implementação**

1. **Buscar e Normalizar os Dados**:
   * Buscar os dados de inflação, taxa de juros e PIB para o período de 2014-2024.
   * Normalizar e interpolar esses dados para que fiquem com frequência mensal ou diária, conforme necessário.
2. **Feature Engineering com os Indicadores**:
   * Incluir **médias móveis** e **volatilidade** desses indicadores para capturar mudanças recentes e tendências.
   * Incorporar esses dados ao conjunto de treino como variáveis independentes.

Coleta desses dados do Banco Central e ajustarei o modelo para incluir essas variáveis. Logo após, aplicaremos os hiperparâmetros otimizados e realizaremos novos testes para analisar os impactos.

### Turbinando o modelo LSTM

**Passos que Seguiríamos para Construir Novos Modelos**

1. **Preparação dos Dados**:
   * **Normalização e Escalonamento**: Aplicar MinMaxScaler para garantir que os dados estejam na mesma escala.
   * **Alinhamento Temporal**: Manter a consistência temporal entre dados financeiros e macroeconômicos (uso de resampling diário e forward-fill para dados macroeconômicos).
   * **Transformação em Sequências**: Preparar as janelas de dados com sequence\_length fixo para o modelo LSTM, organizando os dados em X (sequências) e y (target).
2. **Definição de Hiperparâmetros**:
   * **Grid de Hiperparâmetros**: Definir um grid de parâmetros que queremos testar, incluindo unidades LSTM, taxa de dropout, batch size, e épocas.
   * **Estratégia de Validação**: Usar validação cruzada com TimeSeriesSplit para garantir que o modelo seja testado em dados futuros sem vazamento de informação.
3. **Busca de Hiperparâmetros**:
   * **Execução da Busca Manual**: Testar as combinações de hiperparâmetros definidas no grid e calcular o RMSE médio para cada uma.
   * **Seleção dos Melhores Parâmetros**: Escolher a combinação que resulta no menor RMSE médio como a melhor configuração para o modelo.
4. **Treinamento Final com os Melhores Hiperparâmetros**:
   * **Treinamento Completo**: Treinar o modelo com todos os dados disponíveis usando os melhores hiperparâmetros encontrados na busca.
   * **Avaliação**: Fazer previsões para os últimos dias do conjunto de dados e avaliar com as métricas RMSE, MAE, e MAPE.
5. **Visualização dos Resultados**:
   * **Gráfico de Previsões vs. Valores Reais**: Visualizar as previsões do modelo em comparação com os valores reais.
   * **Tabela de Métricas**: Exibir as métricas de erro para os últimos 30 dias e, se necessário, para cada fold.

**Próximos Algoritmos e Possíveis Melhorias**

1. **Modelos Mais Complexos**:
   * Podemos testar arquiteturas mais complexas, como **pilhas de LSTM** ou mesmo **Bidirectional LSTM**, se acharmos que a estrutura de série temporal exige uma captura de contexto mais robusta.
   * **Redes GRU** (Gated Recurrent Units) são outra alternativa interessante para explorar, pois são computacionalmente mais eficientes que LSTM e podem oferecer desempenho semelhante.
2. **Adição de Mais Indicadores Macroeconômicos**:
   * Expandir o dataset com outros indicadores macroeconômicos ou de mercado (ex.: taxa de câmbio, índices de volatilidade, etc.), o que pode melhorar o contexto para previsões financeiras.
3. **Hiperparâmetros Otimizados**:
   * **Otimização Bayesiana** para a busca de hiperparâmetros em vez da busca manual, o que pode encontrar a melhor configuração de forma mais rápida e precisa.
   * **Ajuste de batch\_size e epochs** baseado na observação da convergência do modelo.
4. **Experimentação com Diferentes Estruturas de Seqüências**:
   * **Ajuste do sequence\_length**: Testar diferentes comprimentos de sequência para entender qual proporciona o melhor contexto temporal.
   * **Variáveis-alvo Alternativas**: Experimentar previsões para outros ativos ou mesmo múltiplos ativos como alvos (multitarefa).
5. **Integração de Outros Modelos**:
   * **Modelos de atenção**: Integrar camadas de atenção ao LSTM para melhorar a capacidade do modelo de focar em períodos relevantes.
   * **Modelos híbridos com CNN**: Para capturar padrões temporais e espaciais, como flutuações sazonais curtas e tendências.
6. **Comparação de Resultados e Benchmarking**:
   * Acompanhar métricas de desempenho para cada modelo desenvolvido e criar um benchmark de comparação entre as diferentes arquiteturas e estratégias de otimização.
   * Manter um histórico para saber quais abordagens funcionam melhor em diferentes tipos de cenários.

**Documentação e Procedimentos para Reprodutibilidade**

Para facilitar o trabalho contínuo e a reprodutibilidade dos experimentos, é importante:

* **Documentar o código**: Adicionar comentários e descrições detalhadas nos scripts, especialmente em pontos importantes como a definição de hiperparâmetros e o fluxo de dados.
* **Registro de Hiperparâmetros e Resultados**: Usar um arquivo ou planilha para registrar todas as combinações de hiperparâmetros testadas e suas métricas de desempenho.
* **Pipeline Automatizado**: À medida que avançamos, podemos estruturar um pipeline de experimentos que rode sequencialmente várias configurações de modelos e armazene os resultados.

### Estratégia de Curto e Longo Prazo, roteiro completo

**Estratégia de Modelagem com Horizonte de Previsão de Curto e Longo Prazo**

Com um histórico extenso e um horizonte de previsão de médio a longo prazo, é crucial adotar uma abordagem que segmente os objetivos e use modelos especializados para capturar as características temporais e complexidades em diferentes escalas temporais.

**1. Divisão dos Objetivos:**

* **Curto Prazo (30-90 dias)**:
  + Focar em capturar a volatilidade e as oscilações diárias. Esse período é influenciado por fatores de curto prazo, como eventos econômicos e volatilidade de mercado.
  + **Modelos Indicados**:
    - **LSTM Multivariado**: Ideal para capturar relações entre múltiplas variáveis em sequência temporal.
    - **Modelos Baseados em Árvores (XGBoost, LightGBM)**: São robustos para capturar padrões complexos e podem ser ajustados para dados temporais em intervalos mais curtos.
    - **Ajustes e Pré-Processamento**: Usar janelas temporais mais curtas (ex.: 30 a 60 dias) e adicionar variáveis que indiquem eventos recentes ou volatilidade.
* **Médio e Longo Prazo (1-2 anos)**:
  + O objetivo é capturar tendências e sazonalidades, além de considerar eventos econômicos cíclicos e choques externos que podem impactar o mercado.
  + **Modelos Indicados**:
    - **ARIMA e Prophet**: Modelos que capturam bem tendências e sazonalidades, além de permitirem ajustes para eventos sazonais.
    - **LSTM Seq2Seq**: Uma variação do LSTM que mapeia sequências para previsões futuras em períodos mais longos.
    - **Modelos GARCH**: Para modelar e prever a volatilidade, especialmente útil em séries financeiras onde a volatilidade tende a ser autocorrelacionada.
    - **Ensemble Final**: Combinar previsões de diferentes modelos para obter maior robustez e reduzir o risco de super ou subajuste em apenas um modelo.

**2. Implementação da Estratégia de Modelagem**

* **Etapas para Modelagem de Curto Prazo**:
  + **Preparação de Dados**: Segmentar a base em janelas curtas e escalonar as variáveis com MinMaxScaler.
  + **Ajuste de Modelos de Curto Prazo**: Treinar e ajustar hiperparâmetros de LSTM multivariado e XGBoost, focando em previsões de curto alcance (ex.: próximas 30-90 dias).
  + **Validação Temporal**: Validar os modelos com divisão temporal (TimeSeriesSplit) para garantir que estamos testando em dados "futuros".
* **Etapas para Modelagem de Médio e Longo Prazo**:
  + **Identificação de Padrões de Sazonalidade e Tendência**: Usar ARIMA e Prophet para identificar e prever componentes sazonais e de tendência.
  + **Aplicação de Modelos Seq2Seq e GARCH**: Implementar LSTM Seq2Seq para prever séries temporalmente correlacionadas e GARCH para modelagem de volatilidade de longo prazo.
  + **Combinação de Previsões (Ensemble)**: Utilizar ensemble para combinar previsões dos modelos ARIMA, Prophet, LSTM Seq2Seq e GARCH para maior precisão e estabilidade.

**3. Pipeline de Modelagem e Validação Final**

* **Integração dos Modelos em um Pipeline**: Automatizar o fluxo de dados e treinar modelos com divisões apropriadas para curto e longo prazo.
* **Avaliação e Seleção de Modelos**:
  + Comparar métricas de desempenho (RMSE, MAE, MAPE) em janelas de curto e longo prazo para identificar os modelos que oferecem melhor precisão para cada horizonte.
* **Implementação do Ensemble Final**: Integrar as previsões dos diferentes modelos e ponderar cada previsão com base na performance histórica e no horizonte de previsão.

**Conclusão**

Com essa estratégia, conseguiremos capturar nuances temporais tanto de curto quanto de longo prazo, aumentando a robustez da previsão e permitindo um entendimento mais profundo dos diferentes períodos temporais. Amanhã, podemos iniciar com a implementação e o ajuste dos modelos para cada horizonte, garantindo uma base sólida para previsões de médio e longo prazo.

—————————————29/10/2024——————————————————

### Final dos testes com LSTM - curto e médio prazo

Finalizado o algoritmo t20\_script\_LSTM\_mult\_hip.py

### Trabalhando com métodos de médio e longo prazo

**Estrutura para o Algoritmo de Médio e Longo Prazo (2 a 5 anos)**

**Objetivo**

**Capturar:**

Tendências e sazonalidades no mercado, que são mais evidentes em horizontes mais longos.

Eventos cíclicos e choques econômicos que podem ter efeitos prolongados.

Volatilidade autocorrelacionada em séries financeiras (picos de volatilidade em períodos próximos).

**Estrutura de Modelagem Proposta**

**Modelos para Capturar Tendências e Sazonalidades:**

1. **ARIMA e Prophet:** ótimos para detectar tendências de longo prazo e padrões sazonais. O Prophet, em particular, permite incorporar feriados, eventos externos e outros fatores que possam afetar sazonalidades. Ajuste dos Parâmetros: Precisamos fazer testes para encontrar os melhores parâmetros, considerando sazonalidades anuais e semestrais.
2. **Modelos Seq2Seq para LSTM:**

LSTM Seq2Seq: diferente do LSTM multivariado simples, essa variação permite criar previsões mais adequadas para períodos mais longos ao mapear sequências inteiras de entrada para sequências de saída.

Estrutura do Seq2Seq: utilizaremos camadas de codificador e decodificador, onde o codificador processa uma janela de tempo e o decodificador mapeia essa janela para o período futuro. Isso permitirá capturar dependências de longo alcance nas séries temporais.

1. **Modelos GARCH para Volatilidade:**

GARCH (Generalized Autoregressive Conditional Heteroskedasticity): Ideal para modelar a volatilidade de longo prazo, especialmente em mercados financeiros, onde a volatilidade tende a ser autocorrelacionada.

Integração de Volatilidade: podemos usar o GARCH para gerar uma série de previsões de volatilidade e adicionar essa informação como uma entrada auxiliar no modelo LSTM Seq2Seq ou nos modelos ARIMA e Prophet.

1. **Ensemble de Modelos para Robustez:**

Ensemble: combinar as previsões dos diferentes modelos (ARIMA, Prophet, LSTM Seq2Seq e GARCH) em um ensemble. A estratégia mais comum é uma média ponderada, mas podemos usar uma regressão para ajustar os pesos.

Vantagem do Ensemble: a combinação ajudará a capturar os diferentes aspectos da série temporal, melhorando a robustez e reduzindo a chance de super ou subajuste de apenas um modelo.

**Implementação em Dois Algoritmos Distintos**

1. Algoritmo de Curto Prazo (já desenvolvido):

Focado em previsões de curto prazo com LSTM multivariado, ajustado para capturar oscilações diárias ou semanais.

2. Algoritmo de Médio e Longo Prazo:

Novo Algoritmo: vamos implementar um novo pipeline, focado exclusivamente em previsões de médio a longo prazo. Esse novo algoritmo terá uma estrutura modular para os diferentes modelos (ARIMA, Prophet, LSTM Seq2Seq e GARCH).

Configuração Modular: cada modelo terá seu próprio módulo para otimização e ajuste, permitindo um ensemble final para previsões robustas.

**Próximos Passos**

Implementação de Modelos para Médio e Longo Prazo:

Construir módulos separados para ARIMA, Prophet, LSTM Seq2Seq e GARCH.

Testar individualmente para entender a contribuição de cada um ao conjunto.

**Ensemble de Previsões:**

Após obter previsões individuais, implementar uma função de ensemble para combinar os resultados.

**Avaliação e Métricas:**

Avaliar cada modelo e o ensemble usando RMSE, MAE, e MAPE para assegurar que o conjunto fornece uma melhoria significativa.

**Estrutura Comum para Todos os Algoritmos**:

Este código fornece uma estrutura comum para o pipeline, contendo:

Carregamento e preparação dos dados (prepare\_data), com a preparação de frequências mensais e preenchimento de valores ausentes.

Módulos para cada tipo de modelo (Prophet, ARIMA, GARCH, e LSTM Seq2Seq), onde cada modelo pode ser executado e avaliado individualmente.

Ensemble Final (ensemble\_forecast), combinando as previsões de diferentes modelos para obter uma previsão mais robusta.

**Ponto de Partida para Cada Algoritmo:**

Usaremos este código como "base" para cada modelo de machine learning que planejamos testar. Cada modelo tem seu próprio bloco de código específico, mas a estrutura geral permanece a mesma.

Quando testarmos um novo algoritmo ou ajustarmos hiperparâmetros, faremos isso dentro desta estrutura, substituindo ou ajustando o módulo específico para aquele modelo.

**Modularidade** **e** **Reusabilidade**:

Como o código é modular, ele facilita a adição e a troca de algoritmos. Podemos desenvolver e testar cada modelo (Prophet, ARIMA, GARCH, LSTM Seq2Seq) separadamente e, depois, integrar facilmente ao pipeline final.

Cada componente é encapsulado em funções (prophet\_model, arima\_model, garch\_model, lstm\_seq2seq\_model, etc.), permitindo que ajustemos um modelo sem impactar os outros.

**Refinamento Iterativo e Ensemble:**

Após testar e validar cada modelo individualmente, refinaremos o ensemble. A função ensemble\_forecast agora usa uma média simples das previsões. Em etapas futuras, podemos ajustar o ensemble com combinações ponderadas ou algoritmos de regressão, usando os resultados individuais dos modelos.

Isso também nos permite adicionar ou remover modelos do ensemble facilmente, dependendo de seu desempenho.

**Exemplo de Iteração no Pipeline**

Se decidirmos testar um novo modelo, como XGBoost para prever a tendência de longo prazo, por exemplo, podemos:

Adicionar uma função xgboost\_model ao código para treinar e prever usando XGBoost.

Inserir as previsões de XGBoost no ensemble junto com os outros modelos.

Testar se o ensemble melhora com esse novo modelo.

## Análise algoritmo t20\_script\_pipeline\_arima\_med\_longo.py

1. **Estrutura e Modularidade**:

* O código está bem estruturado, com funções individuais para cada modelo (Prophet, ARIMA e GARCH), além de uma função para o ensemble. Isso permite modularidade e reusabilidade, alinhado com o que discutimos.

1. **Modelos Específicos para Captura de Características**:

* **Prophet**: Está configurado para capturar sazonalidade anual e mensal, o que é ideal para detectar padrões de longo prazo.
* **ARIMA**: Ajustado para capturar tendências no horizonte de 2 a 5 anos.
* **GARCH**: Implementado para prever volatilidade dinâmica, adicionando uma camada de análise de risco com base na volatilidade.

1. **Ensemble com Volatilidade Dinâmica**:

* A combinação entre o Prophet e o ARIMA com ajuste pela volatilidade de GARCH está implementada, e essa abordagem é muito eficaz para combinar tendências e volatilidade, produzindo uma previsão robusta.

1. **Visualização e Tabela de Resultados**:

* A visualização está bem formatada, mostrando as previsões dos diferentes modelos e o ensemble final. A tabela à direita oferece um resumo claro dos valores do ensemble.

### Inserção de Simulação de Monte Carlo no processo 🡪 t20\_scriot\_monte\_carlo\_simulation.py

**Objetivo desse algoritmo**: realizar a simulação de Monte Carlo.

**Estrutura do Algoritmo**

Aqui está o que será incluído no código:

1. Carregamento das Previsões do Ensemble:
2. Importar as previsões do ensemble que foram geradas pelo código anterior. Vamos considerar as previsões de médio e longo prazo como a linha de base para a simulação.
3. Simulação de Monte Carlo:

* Implementar o método de Monte Carlo, que simula várias trajetórias futuras usando uma distribuição de retornos com base na média e desvio padrão observados nos dados históricos.
* Definir um número de simulações (ex.: 1000) e um horizonte de tempo de 2 a 5 anos.
* Para cada simulação, usar a previsão do ensemble como base e adicionar variação de acordo com a volatilidade projetada.

1. Visualização dos Resultados:

* Plotar as trajetórias simuladas sobre a previsão do ensemble.
* Gerar uma tabela que exiba os percentis das projeções para ajudar na análise (exemplo: percentis 5%, 50%, e 95%).

## Próximas Etapas 🡪 caminhando para a conclusão do projeto

1. **Integração com Análises de Risco e Retorno**:

* Agora que temos as trajetórias simuladas, podemos calcular métricas adicionais, como o **Value at Risk (VaR)** e o **Expected Shortfall (ES)** para diferentes níveis de confiança. Isso fornecerá insights sobre os riscos financeiros envolvidos.

1. **Implementação de um Dashboard Interativo**:

* Podemos considerar a criação de um **dashboard interativo** (usando ferramentas como Plotly Dash ou Streamlit) para explorar o modelo e os resultados de Monte Carlo em tempo real. Isso permitirá que o usuário ajuste os parâmetros (número de simulações, volatilidade, limites) e visualize as mudanças de forma dinâmica.

1. **Backtesting e Validação do Modelo**:

* Realizar uma análise de backtesting usando dados históricos para verificar o quão bem o modelo performa em cenários passados. Esse teste de robustez é fundamental para garantir que o modelo seja confiável para previsões reais.

1. **Automação do Processo de Simulação e Relatórios**:

* Podemos automatizar o processo de execução da simulação e geração de relatórios periódicos com os resultados do Monte Carlo, incluindo as métricas de desempenho e riscos. Esse pipeline automatizado pode ser útil em uma aplicação prática.

1. **Preparação para Tomada de Decisões**:

* Finalmente, com os dados da simulação prontos, podemos integrá-los em um sistema de suporte à decisão. Isso poderia envolver definir alertas ou gatilhos baseados em limites de risco aceitáveis para ajudar na tomada de decisões financeiras.

### Refinamento das propostas para incluir um portfólio otimizado capaz de superar o índice BOVESPA

## Etapas Revisadas para Construção e Otimização de Portfólio

1. **Definir Universo de Ativos e Restrições de Peso**:
   * **Objetivo**: Selecionar um conjunto de ativos para o portfólio (ações ou ETFs) que servirão como componentes.
   * **Restrição de Peso**: Impor limites de peso por ativo para garantir que o portfólio seja balanceado (ex: mínimo de 15% e máximo de 60% por ativo).
   * **Dados Necessários**: Dados históricos de preços de cada ativo do portfólio e do índice BOVESPA para calcular o retorno esperado e a volatilidade, bem como para medir a correlação entre os ativos.
2. **Simulação de Monte Carlo Ajustada para o Portfólio**:
   * **Objetivo**: Utilizar Monte Carlo para simular cenários para o portfólio agregado e ver como ele performa comparado ao BOVESPA.
   * **Refinamento**: Em vez de simular cada ativo isoladamente, simularemos o portfólio com base nos retornos ponderados dos ativos. O Monte Carlo agora servirá para explorar o impacto da diversificação e alocação de ativos sob condições de mercado incertas.
   * **Benchmarking**: Comparar o desempenho do portfólio simulado com o índice BOVESPA, calculando o tracking error, que é a diferença entre o retorno do portfólio e o do benchmark.
3. **Otimização de Pesos (Alocação)**:
   * **Objetivo**: Encontrar a melhor combinação de pesos para maximizar o retorno ajustado ao risco, visando superar o índice BOVESPA.
   * **Método**: Utilizar algoritmos de otimização (como a otimização de média-variância de Markowitz) com as restrições de peso definidas.
   * **Backtesting e Validação**: Validar a alocação com dados históricos do índice BOVESPA, assegurando que a carteira supera consistentemente o índice. Isso ajuda a entender como a carteira se comportaria em cenários históricos.
4. **Cálculo de Taxa de Retorno Anualizada e Outras Métricas**:
   * **Objetivo**: Medir a performance do portfólio em termos de taxa de retorno anualizada, volatilidade, Sharpe ratio, e beta em relação ao BOVESPA.
   * **Análise de Desempenho**: Com o objetivo de superar o benchmark, podemos acompanhar métricas como:
     + **Sharpe Ratio**: Para avaliar o retorno ajustado ao risco.
     + **Alfa e Beta em Relação ao BOVESPA**: Alfa para ver o desempenho excessivo em relação ao benchmark e beta para medir a sensibilidade ao índice.
     + **Tracking Error**: Para ver o desvio do portfólio em relação ao BOVESPA.
5. **Dashboard e Automação**:
   * **Visualização**: Expandir o dashboard para incluir visualizações da alocação ótima, desempenho comparado ao índice, e o retorno histórico do portfólio ajustado para Monte Carlo.
   * **Automação**: Criar um processo automatizado para rodar a otimização de portfólio em intervalos regulares, acompanhando o desempenho e gerando relatórios comparativos com o índice BOVESPA.

### Implementação e Ferramentas

Para implementar esses passos, sugiro algumas ferramentas e técnicas:

* **Otimização com Restrições**: Utilizar bibliotecas como scipy.optimize para configurar uma função de custo que maximiza o retorno ajustado ao risco com restrições de peso.
* **Simulação e Análise de Portfólio**: numpy e pandas continuarão sendo úteis para cálculos de retorno e risco, enquanto plotly será útil para visualizações dinâmicas do portfólio.
* **Métricas de Benchmarking**: numpy para calcular métricas de risco e desempenho e scipy.stats para o tracking error e alfa em relação ao BOVESPA.

### Exemplos de Métricas e Benchmark

* **Taxa de Retorno Anualizada**: ((1 + retorno\_acumulado)\*\*(1/anos)) - 1
* **Tracking Error**: Desvio padrão da diferença de retornos do portfólio e do benchmark.
* **Sharpe Ratio e Beta**: Relacionados ao BOVESPA, calculados com base nas séries temporais de retornos.

### Estrutura sugerida para condução do projeto

Seguir um processo modular de desenvolvimento será a melhor abordagem, pois isso nos permitirá:

1. **Desenvolver e Testar Cada Componente Individualmente**: Criar um algoritmo para cada etapa e validar cada um em separado nos ajuda a identificar problemas específicos e a garantir que cada parte funcione conforme esperado antes de integrá-las.
2. **Facilidade de Depuração e Manutenção**: Com cada componente isolado, é mais fácil fazer melhorias, ajustar parâmetros e otimizar o código sem afetar o sistema completo.
3. **Facilidade para Futuras Expansões**: Uma estrutura modular permitirá adicionar mais funcionalidades (como otimizações de portfólio, novos benchmarks, ou outros indicadores) de forma mais simples e organizada.

### Estrutura Modular Sugerida

Vamos dividir o projeto nas seguintes fases modulares, onde cada módulo será um arquivo de código ou uma função autossuficiente. No final, integramos cada módulo em um sistema completo e robusto.

#### 1. Módulo de Coleta e Preparação de Dados

* **Objetivo**: Reunir os dados históricos dos ativos escolhidos para o portfólio e do índice BOVESPA.
* **Funcionalidades**:
  + Carregar os dados de preços de fechamento ajustados para cada ativo e para o índice BOVESPA.
  + Preprocessamento dos dados (limpeza, ajuste de datas, cálculo de retornos).
* **Teste**: Validar se todos os dados estão no formato correto e se o cálculo dos retornos está preciso.

#### 2. Módulo de Simulação de Monte Carlo para Cada Ativo

* **Objetivo**: Implementar a simulação de Monte Carlo ajustada para cada ativo e agregar as trajetórias simuladas.
* **Funcionalidades**:
  + Simular cenários de preços para cada ativo com base em sua volatilidade histórica e retorno médio.
  + Avaliar o efeito de volatilidade sobre o portfólio de forma individual.
* **Teste**: Visualizar as trajetórias simuladas para verificar a variação esperada e ajustar a volatilidade, se necessário.

#### 3. Módulo de Otimização de Portfólio (Alocação de Pesos)

* **Objetivo**: Determinar a melhor combinação de pesos para maximizar o retorno ajustado ao risco, com restrições de peso para cada ativo.
* **Funcionalidades**:
  + Implementar um modelo de otimização de média-variância, usando scipy.optimize, considerando os limites mínimo e máximo de peso para cada ativo.
  + Otimizar o portfólio para minimizar o tracking error em relação ao BOVESPA, mas ao mesmo tempo superar o benchmark.
* **Teste**: Verificar a composição do portfólio e visualizar o impacto das diferentes alocações sobre o retorno e o risco.

#### 4. Módulo de Simulação de Monte Carlo para o Portfólio Otimizado

* **Objetivo**: Usar a alocação otimizada para simular o portfólio agregado e projetar o desempenho do portfólio no futuro.
* **Funcionalidades**:
  + Aplicar a simulação de Monte Carlo ao portfólio ponderado para observar como ele se comporta em diferentes cenários de mercado.
  + Comparar o desempenho projetado com o benchmark.
* **Teste**: Visualizar as projeções do portfólio agregado e comparar com o BOVESPA para verificar se o portfólio está, de fato, superando o benchmark.

#### 5. Módulo de Análise de Desempenho e Métricas de Benchmark

* **Objetivo**: Calcular métricas de desempenho do portfólio, como retorno anualizado, Sharpe Ratio, beta em relação ao BOVESPA, tracking error, e alfa.
* **Funcionalidades**:
  + Calcular métricas de desempenho comparativas, incluindo análise de erro e volatilidade.
  + Incluir visualizações e uma tabela para apresentar as métricas de desempenho.
* **Teste**: Avaliar os cálculos e assegurar que as métricas estão em linha com o objetivo de superar o benchmark.

#### 6. Módulo de Dashboard e Relatório

* **Objetivo**: Integrar todos os módulos em um dashboard visual e interativo para análise e acompanhamento.
* **Funcionalidades**:
  + Apresentar gráficos, tabelas e métricas para cada ativo, o portfólio agregado, e o índice BOVESPA.
  + Gerar relatórios automatizados com resumo das métricas, comparações com o benchmark e previsões de desempenho.
* **Teste**: Validar a visualização do dashboard e garantir que os relatórios sejam gerados corretamente.

#### Exemplo de Fluxo de Integração

Ao final de cada etapa, integramos os módulos em um fluxo completo:

1. **Etapa 1 + 2**: Coletar dados e rodar a simulação de Monte Carlo inicial para entender a variabilidade de cada ativo.
2. **Etapa 1 + 3**: Coletar dados e otimizar a alocação de portfólio.
3. **Etapa 3 + 4**: Aplicar a simulação de Monte Carlo no portfólio otimizado e comparar com o benchmark.
4. **Etapa 5**: Calcular métricas de desempenho.
5. **Etapa 6**: Apresentar tudo no dashboard final, permitindo ajustes e análises periódicas.

#### Ferramentas e Repositório

Recomendo que versionemos o código em um repositório (como GitHub) para facilitar o controle de versões, colaborarmos nas etapas e documentarmos cada módulo. Cada módulo pode ser um script ou um notebook separado, para testes individuais, e no final integramos tudo em um único sistema ou notebook final com a execução automatizada.

#### Próximos Passos

1. **Definir o universo de ativos** para o portfólio.
2. **Desenvolver o primeiro módulo** (coleta e preparação dos dados) e seguir para o segundo módulo após validação.
3. Testar cada etapa e ajustar, conforme necessário.

## Passos para a conclusão

### Passo 1: Revisão e Integração dos Algoritmos Existentes

1. **Coleta de Dados**:
   * Analisar e consolidar os scripts de coleta de dados econômicos e dados de ativos para garantir que as informações estejam atualizadas e em um formato consistente para o restante dos módulos.
2. **Testes de Consistência dos Dados**:
   * Integrar o script de teste de consistência dos dados. Este módulo deve rodar automaticamente após a coleta de dados, sinalizando qualquer problema de dados antes de iniciar as análises e simulações.
3. **Modelos de Previsão**:
   * Revisar os scripts de previsão com LSTM e ARIMA. Esses modelos já são fundamentais para a análise de cada ativo no portfólio e fornecem previsões que servem de base para a simulação de Monte Carlo e o cálculo do retorno projetado.
   * Modularizar os modelos para que cada ativo tenha previsões independentes e possam ser agregadas no portfólio, com flexibilidade para ajuste de parâmetros específicos de cada ativo.
4. **Simulação de Monte Carlo**:
   * Incorporar o script de Monte Carlo, agora ajustado com volatilidade, percentis, e cálculo de erro. Este módulo deve receber as previsões dos modelos (LSTM e ARIMA) e rodar cenários que ajudam a entender as variações e riscos esperados para o portfólio.

### Passo 2: Desenvolvimento de Módulos Adicionais para o Portfólio

1. **Cálculo de Retorno e Volatilidade do Portfólio**:
   * Criar um módulo para calcular o retorno anual e a volatilidade com base nas simulações de Monte Carlo e nos resultados dos modelos de previsão. Esse módulo consolidará o desempenho esperado do portfólio.
2. **Otimização de Pesos do Portfólio**:
   * Desenvolver um algoritmo para otimizar os pesos de cada ativo no portfólio, respeitando os limites de alocação (mínimo 15% e máximo 60%) e visando maximizar o retorno ajustado ao risco em relação ao índice Bovespa.
   * Considerar a implementação de algoritmos de otimização, como a Otimização por Fronteira Eficiente ou algoritmos evolutivos, para equilibrar o portfólio de forma robusta.
3. **Comparação com o Benchmark (Índice Bovespa)**:
   * Incluir um módulo para calcular o retorno do Bovespa no mesmo período, usando dados históricos e projetados. Esse módulo permitirá uma comparação direta entre o desempenho do portfólio e o benchmark.

### Passo 3: Integração e Automação do Sistema Completo

1. **Orquestração**:
   * Integrar todos os módulos em uma pipeline de execução, onde cada etapa se alimenta automaticamente da saída da anterior. Isso garantirá que, ao coletar novos dados, todo o fluxo (coleta, teste, previsão, simulação, otimização e comparação) seja executado de forma sequencial.
2. **Interface para Visualização de Resultados**:
   * Desenvolver uma interface para visualizar as métricas do portfólio, os gráficos de Monte Carlo, os retornos esperados, os pesos dos ativos e a comparação com o índice Bovespa. A interface pode ser construída com um framework como Streamlit ou Plotly Dash, para facilitar o acompanhamento e análise em tempo real.
3. **Sistema de Alertas e Relatórios**:
   * Implementar alertas automáticos para qualquer anomalia detectada durante a coleta de dados ou nos testes de consistência, bem como relatórios periódicos do desempenho do portfólio em comparação ao Bovespa.

### Próximos Passos

1. **Revisar e testar cada módulo individualmente, com especial atenção para os pontos de integração entre os módulos existentes e os novos**.
2. **Executar testes com o portfólio usando diferentes combinações de ativos e pesos** para ajustar os parâmetros da otimização e garantir que o portfólio cumpre o objetivo de superar o benchmark.
3. **Criar scripts de monitoramento e agendamento** para rodar o sistema automaticamente em intervalos de tempo definidos, coletando e analisando os dados atualizados.

Essa abordagem modular garante flexibilidade e escalabilidade ao projeto. A cada módulo finalizado e testado, teremos uma parte do sistema funcional, facilitando o desenvolvimento e a validação da etapa seguinte.