**PROJETO APLICADO 4: SÉRIES TEMPORAIS – ENERGIA**

**Grupo Projeto Aplicado:**

**ISABEL DE FÁTIMA BATANETE RAMOS - 10056372**

**MAIARA DE SALES FAGUNDES RODRIGUES – 10407555**

**RENAN MOREIRA PEREIRA – 10407030**

**SÃO PAULO**

**2024**

**SUMÁRIO**

**PARTE I**

1.Resumo/Abstract.............................................................................................................................................3

2. Introdução.......................................................................................................................................................4

3. Referencial teórico..........................................................................................................................................7

4.Definição da linguagem de programação usada no projeto ...........................................................................11

5.Pipeline e Metodologia ....................................................................................................................................9

6.Descrição da base de dados..........................................................................................................................18

7.Análise exploratória da base de dados escolhida..........................................................................................19

8.Tratamento da base de dados, Preparação e treinamento. .........................................................................20

9.Resultados Esperados ..................................................................................................................................21

10.Conclusões e Trabalhos Futuros.................................................................................................................22

11.Referências Bibliográficas...........................................................................................................................23

12.Cronograma de atividades..........................................................................................................................25

13.Github e YouTube........................................................................................................................................27

**Resumo**

Este projeto investiga a aplicação de técnicas de séries temporais para a previsão da geração e consumo de energia renovável, com foco na Grécia, alinhando-se ao Objetivo de Desenvolvimento Sustentável (ODS) 7 da ONU, que busca garantir o acesso universal à energia limpa e acessível. Utilizando a base de dados pública sobre a geração de energia solar e eólica na Grécia. O estudo visa desenvolver modelos preditivos com base em algoritmos como Prophet e métodos baseados em árvores, para otimizar o uso de energia renovável e mitigar a dependência de fontes não renováveis. A previsão precisa desses padrões ajudará na formulação de políticas públicas e na gestão energética de forma mais eficiente, permitindo a integração de fontes intermitentes como a solar e a eólica.

**Abstract**

This project explores the application of time series forecasting techniques for analyzing renewable energy generation and consumption in Greece, in alignment with the UN's Sustainable Development Goal (SDG) 7, which promotes universal access to clean and affordable energy. Using publicly available datasets on solar and wind energy in Greece. The study aims to develop predictive models leveraging algorithms like *Prophet* and tree-based ensemble methods to optimize renewable energy utilization and reduce reliance on non-renewable sources. Accurate forecasting of energy patterns will aid in strategic decision-making for public policies and energy management, especially in integrating intermittent sources like solar and wind into the grid.

**Introdução**

O desafio de equilibrar o fornecimento de energia e a sustentabilidade ambiental é uma preocupação crescente em todo o mundo. Com a crise climática e a necessidade de reduzir as emissões de gases de efeito estufa, as energias renováveis se destacam como a principal solução para descarbonizar a matriz energética. A Grécia, com sua alta incidência solar, e o Brasil, com uma matriz elétrica já predominantemente renovável, são exemplos de como diferentes regiões estão se movendo em direção a um futuro energético mais sustentável.

De acordo com a Agência Internacional de Energia (AIE), as energias renováveis representaram 29% da geração global de eletricidade em 2021, sendo a energia solar responsável por 11% dessa participação. Na Grécia, a capacidade instalada de energia solar fotovoltaica atingiu 4 GW em 2022, representando 11% da matriz energética total do país (Hellenic Association of Photovoltaic Companies, 2023). Em comparação, o Brasil, líder em energias renováveis, tinha cerca de 16,4 GW de capacidade instalada de energia solar até o final de 2022, o que representava cerca de 9,5% da capacidade instalada do país (ANEEL, 2023). Esse crescimento acelerado reflete o papel fundamental que a energia solar está desempenhando na diversificação das fontes energéticas e na mitigação das mudanças climáticas.

A Grécia, com sua geografia ensolarada, está em uma posição estratégica para expandir sua capacidade de energia solar. O governo grego estabeleceu metas ambiciosas, com planos para que 35% da eletricidade gerada no país venha de fontes renováveis até 2030. Entre as fontes de energia renovável, a solar é considerada a mais promissora devido à sua abundância. De fato, a Grécia é uma das regiões da Europa com maior índice de insolação, variando entre 1.500 a 1.900 kWh/m² anualmente, dependendo da localização.

De acordo com a AIE, a Grécia está aumentando sua capacidade de armazenamento de energia para lidar com a variabilidade inerente da geração solar. O país pretende expandir a capacidade de armazenamento em 10 GW até 2030 para garantir a segurança no fornecimento energético durante períodos de baixa geração, como à noite ou em dias nublados. Segundo a International Renewable Energy Agency, 2023, estudos indicam que a utilização de baterias associadas a painéis solares pode melhorar a eficiência do sistema e reduzir as emissões de carbono em até 20%.

O Brasil, por outro lado, já tem uma das matrizes energéticas mais limpas do mundo, com 87% da capacidade de geração elétrica proveniente de fontes renováveis, sendo a hidrelétrica a mais dominante (64%) (ANEEL, 2023). No entanto, a energia solar tem crescido de forma acelerada nos últimos anos. Entre 2018 e 2022, a capacidade instalada de energia solar no Brasil saltou de 2,2 GW para 16,4 GW, um aumento de mais de 640%. Este crescimento é impulsionado tanto por iniciativas governamentais quanto pela adoção da geração distribuída em propriedades privadas, como residências e empresas.

Uma das diferenças fundamentais entre a Grécia e o Brasil no contexto energético é a diversificação das fontes renováveis. Enquanto a Grécia depende fortemente da energia solar e eólica, o Brasil tem uma matriz mais diversificada, com grande presença da energia hidrelétrica. No entanto, ambos os países enfrentam o desafio de gerenciar a variabilidade de suas fontes renováveis. No caso do Brasil, essa variabilidade é mitigada pela complementaridade entre as fontes hídricas e solares — durante os períodos de estiagem, há maior incidência solar, equilibrando a geração de energia.

A utilização de modelos preditivos de séries temporais é fundamental para otimizar a geração e o consumo de energia renovável. Em países como a Grécia, onde a energia solar desempenha um papel crítico, a previsão da geração em dias e semanas futuros é essencial para garantir a estabilidade da rede elétrica. O artigo "Short-term Renewable Energy Forecasting in Greece using Prophet Decomposition and Tree-based Ensembles" explora o uso de técnicas como Prophet e métodos baseados em árvores para melhorar a precisão das previsões de geração de energia solar e eólica, essenciais para o planejamento energético.

Por exemplo, a implementação de modelos preditivos na Grécia pode ajudar os operadores a prever momentos de baixa geração solar e ativar outras fontes de backup, como baterias ou gás natural, para garantir o abastecimento contínuo. Isso é particularmente importante em países onde a demanda por eletricidade pode flutuar devido a mudanças sazonais e econômicas.

O Brasil, embora com uma matriz elétrica mais equilibrada, também pode se beneficiar dessas técnicas. Modelos preditivos podem ajudar a otimizar a geração distribuída, onde consumidores instalam painéis solares em suas propriedades. Ao prever a produção futura de energia solar, os consumidores podem planejar a venda de excedentes para a rede elétrica, contribuindo para a estabilidade do sistema e maximizando os benefícios financeiros.

Estatisticamente, pode-se mensurar que:

* Grécia: 4 GW de capacidade instalada de energia solar em 2022, com meta de 35% da eletricidade proveniente de fontes renováveis até 2030. A energia solar atualmente representa cerca de 11% da matriz energética.
* Brasil: 16,4 GW de capacidade instalada de energia solar em 2022, com a energia solar representando 9,5% da matriz elétrica. A matriz energética do Brasil é 87% renovável, com destaque para as fontes hidrelétrica, eólica e solar.

Em termos de radiação solar, a Grécia e o Brasil têm níveis de insolação comparáveis. No entanto, o Brasil possui uma vantagem geográfica, com grandes áreas de clima semiárido e baixa densidade populacional, o que permite a instalação de grandes usinas solares em regiões como o Nordeste. No entanto, a gestão da energia solar no Brasil enfrenta desafios em termos de distribuição, especialmente em áreas remotas.

Tanto a Grécia quanto o Brasil enfrentam desafios semelhantes no que diz respeito à integração de grandes volumes de energia solar em suas redes elétricas. A natureza intermitente da energia solar exige soluções de armazenamento eficientes e sistemas de previsão precisos para garantir a segurança do fornecimento. A previsão de consumo, que leva em consideração fatores econômicos, políticos e climáticos, é essencial para equilibrar oferta e demanda e garantir que a eletricidade gerada não seja desperdiçada.

As oportunidades para ambos os países são imensas. A Grécia, com suas metas de energia renovável e incentivos para instalação de painéis solares, pode se tornar um líder europeu na geração solar. Já o Brasil, com sua grande capacidade de geração distribuída, pode continuar expandindo a energia solar como complemento à sua já robusta infraestrutura hidrelétrica.

O principal objetivo deste projeto é desenvolver e implementar modelos preditivos de séries temporais para analisar e prever padrões de geração e consumo de energia renovável. Para isso, foi utilizada a base de dados a partir do repositório fornecido GitHub sob título Greek Solar Wind Energy Forecasting, permitindo a exploração de padrões climáticos e energéticos que impactam a geração solar e eólica na Grécia.

Por fim, mas não menos importante é a motivação para este projeto. Ela está atrelada aos objetivos de carreira dos componentes deste grupo de estudo acadêmico, que visam contribuir para a transição para fontes de energia mais sustentáveis e apoiar políticas públicas que promovam o acesso universal à energia limpa.

**Referencial Teórico**

O presente trabalho se insere na interseção entre o campo de séries temporais e a temática de energia limpa e acessível, em consonância com o Objetivo de Desenvolvimento Sustentável (ODS) 7 das Nações Unidas. O estudo de séries temporais, uma técnica estatística aplicada na modelagem de dados que variam ao longo do tempo, desempenha um papel crucial na análise de padrões históricos e na previsão de eventos futuros, o que é particularmente relevante para a gestão de energia.

1. Séries Temporais

As séries temporais são sequências de dados observadas ao longo do tempo em intervalos regulares. Sua análise é fundamental em diversos campos, como finanças, economia, climatologia e, no contexto deste projeto, na energia. Box e Jenkins (1970) são amplamente reconhecidos por seus estudos iniciais sobre métodos de modelagem de séries temporais, particularmente o modelo ARIMA (Autoregressive Integrated Moving Average). Este modelo, entre outros, é utilizado para identificar padrões e prever valores futuros com base em comportamentos passados.

Em energia, a análise de séries temporais permite a identificação de sazonalidades e tendências no consumo e na geração de energia, fornecendo informações valiosas para a otimização de processos e a tomada de decisões estratégicas. Os modelos de séries temporais ajudam a antecipar picos de demanda, gerenciar a intermitência das fontes renováveis e melhorar a eficiência no uso dos recursos energéticos.

2. Energia Renovável

A energia renovável, derivada de recursos naturais que são constantemente reabastecidos, como solar, eólica, hidráulica e biomassa, tem ganhado importância crescente no cenário global. De acordo com a Agência Internacional de Energia (AIE), as fontes renováveis representaram quase 30% da produção global de eletricidade em 2020. No Brasil, segundo a Empresa de Pesquisa Energética (EPE), mais de 80% da energia elétrica gerada provém de fontes renováveis, com destaque para a hidrelétrica.

No entanto, a variabilidade intrínseca de fontes como solar e eólica exige uma gestão eficaz para garantir a estabilidade da rede elétrica. Nesse sentido, a previsão precisa através de séries temporais é essencial para mitigar os impactos das variações na produção e evitar o desperdício de energia.

3. ODS 7 – Energia Limpa e Acessível

O Objetivo de Desenvolvimento Sustentável 7 (ODS 7) visa garantir acesso a uma energia acessível, confiável, sustentável e moderna para todos até 2030. Este objetivo é parte da Agenda 2030 para o Desenvolvimento Sustentável, estabelecida pela ONU em 2015. A promoção de energia limpa está diretamente ligada à mitigação das mudanças climáticas, redução da pobreza energética e melhoria da qualidade de vida das populações.

Para atingir as metas do ODS 7, é essencial investir em inovação tecnológica e em melhores práticas de gestão energética. Nesse contexto, a aplicação de técnicas de séries temporais para otimizar a geração e o consumo de energia renovável pode contribuir significativamente para alcançar essas metas, garantindo que o fornecimento de energia se torne mais sustentável e eficiente.

4. Aplicação de Modelos Preditivos em Energia

O uso de modelos preditivos na gestão energética é uma tendência crescente, especialmente com o avanço de metodologias baseadas em aprendizado de máquina e inteligência artificial. Métodos como Redes Neurais Artificiais (RNAs), Modelos ARIMA e LSTM (Long Short-Term Memory) são amplamente aplicados em estudos preditivos para energia, oferecendo uma abordagem eficaz para antecipar variações na demanda e na geração de energia.

Estudos de casos específicos, como o de Mena et al. (2018), que aplicaram modelos preditivos na gestão de sistemas de energia eólica, demonstram a viabilidade e a eficácia dessas técnicas em cenários de alta variabilidade. Essas ferramentas fornecem previsões que auxiliam na otimização da operação de usinas de energia renovável, contribuindo para uma rede elétrica mais estável e eficiente.

Além das referências já citadas anteriormente como o de Mena et al. (2018) e Box e Jenkins (1970), em complemento à análise sobre a aplicação de modelos preditivos em séries temporais no campo energético, especialmente no contexto de energias renováveis, baseia-se este trabalho de projeto aplicado em estudos recentes que exploram técnicas variadas para previsão e modelagem, dentre os quais destacam-se os trabalhos de Huang et al. (2024), Shahzad et al. (2023), Benti (2023), Pacella et al. (2024), Zafirakis et al (2023) e Padilha et al. (2022).

Huang et al. (2024) propuseram uma abordagem inovadora para a classificação de séries temporais aplicadas a dispositivos de baixo consumo energético, utilizando iteração de potência atenciosa. Este modelo, voltado para melhorar a eficiência em tempo real, alcançou uma redução de até 70% no consumo de energia em comparação a métodos tradicionais, oferecendo resultados robustos, especialmente para dispositivos com restrições de hardware. Apesar dos bons resultados, a metodologia enfrenta limitações quando aplicada a ambientes com alta volatilidade e rápidas mudanças. Sua aplicabilidade depende de ajustes finos para cenários específicos.

Shahzad et al (2023) realiza uma revisão abrangente dos modelos de previsão de carga de curto prazo (STLF), destacando a importância de abordagens como ARIMA, que utilizam dados históricos para modelar e prever a carga elétrica futura. O estudo ressalta que, embora o modelo ARIMA seja eficaz na captura de padrões sazonais e tendências, ele pode ser limitado pela sua suposição de estacionalidade, necessitando de ajustes para dados não estacionários. Além disso, o uso de redes neurais artificiais (ANNs) tem ganhado destaque por sua capacidade de lidar com relações não lineares complexas, oferecendo maior precisão em previsões quando treinadas com conjuntos de dados robustos. Contudo, as ANNs podem apresentar desafios em termos de interpretabilidade e suscetibilidade a sobre-ajustes.

Um estudo publicado por Benti (2023) realizou uma comparação entre os modelos ARIMA e LSTM (Long Short-Term Memory) para previsão de geração eólica. O ARIMA, amplamente utilizado pela simplicidade e capacidade de lidar com séries estacionárias, apresentou resultados sólidos em dados com comportamentos estáveis. Em contraste, o LSTM, que pode capturar longas dependências temporais, superou o ARIMA em cenários com alta variabilidade. No entanto, o custo computacional elevado do LSTM é uma desvantagem para implementações em larga escala, o que limita sua aplicação em sistemas de tempo real.

Pacella et al. (2024) estudou a Regressão por Processos Gaussianos (GPR) para previsão de carga energética em microgrids, explorando sua capacidade de prever tanto valores futuros quanto as incertezas associadas às previsões. Essa abordagem é particularmente útil para lidar com fontes de energia renováveis intermitentes, como solar e eólica, onde a variabilidade afeta a confiabilidade das previsões. No entanto, a aplicação em grandes volumes de dados pode ser limitada devido ao alto custo computacional da GPR, especialmente quando as séries temporais apresentam alta dimensionalidade Pacella et al. (2024).

A previsão de curto prazo da geração de energia renovável, como discutido por Zafirakis et al (2023) no estudo "Short-term Renewable Energy Forecasting in Greece using Prophet Decomposition and Tree-based Ensembles", destaca-se como um elemento crucial para a integração eficaz de fontes intermitentes, como solar e eólica, nas redes elétricas modernas. Em especial, os autores demonstram o uso de abordagens híbridas, combinando a decomposição do modelo Prophet com métodos baseados em árvores, como Random Forest e Gradient Boosting, para aumentar a precisão nas previsões. Essas técnicas permitem a modelagem de padrões complexos e não lineares presentes nos dados de séries temporais da geração de energia renovável, considerando fatores sazonais, climáticos e econômicos.

O método Prophet, desenvolvido pelo Facebook, é um modelo aditivo capaz de lidar com sazonalidades múltiplas e mudanças de tendência nos dados. Sua flexibilidade em captar variações de longo e curto prazo o torna uma ferramenta apropriada para prever a produção de energia renovável em cenários com alta variabilidade, como a energia solar, que é afetada por condições meteorológicas. No estudo, o modelo foi combinado com métodos de aprendizagem de máquina baseados em árvores, conhecidos por sua robustez ao lidar com grandes volumes de dados e variáveis com interações complexas. Esses métodos permitem não apenas uma maior precisão, mas também oferecem maior interpretabilidade no processo de tomada de decisão em políticas públicas e planejamento energético.

Por fim, Padilha et al. (2022) destacou a importância da aplicabilidade nos modelos de previsão para sistemas energéticos, propondo uma abordagem híbrida que combina técnicas interpretáveis, como árvores de decisão, com modelos preditivos mais complexos, como redes neurais profundas. A pesquisa apontou que modelos mais transparentes são essenciais para a tomada de decisão em cenários críticos, porém há um trade-off entre a precisão e a interoperabilidade, especialmente em sistemas onde a confiabilidade da previsão é crucial para a estabilidade da rede elétrica.

Esses estudos ilustram o potencial das séries temporais para aprimorar a gestão energética, principalmente em fontes renováveis, ajudando a mitigar os desafios inerentes à variabilidade e à intermitência. Cada metodologia possui vantagens e limitações, que devem ser consideradas conforme o contexto de aplicação, seja em previsões de curto ou longo prazo, ou em sistemas com restrições de recursos. Esses trabalhos contribuem significativamente para o avanço da previsão energética, auxiliando na construção de soluções mais sustentáveis e eficientes, em alinhamento com as metas do ODS 7.

Diante a análise dos artigos acima descritos, foi demonstrado que o uso do ARima se mostrou muito utilizado, no entanto, diante da base de dados que foi obtida, julgou-se mais adequado, para o presente trabalho, os modelos propostos por Zafirakis e Padilha com o uso de árvores de decisão. Serão feitos alguns ajustes ao longo das próximas entregas para alcançar o resultado esperado. A Grécia, como caso de estudo, oferece um exemplo concreto de como essas técnicas podem ser aplicadas em países com alta variabilidade na geração de energia solar. A previsão precisa da geração energética é fundamental para garantir a estabilidade da rede elétrica, uma vez que essas fontes intermitentes não podem ser controladas diretamente. Assim, o uso de técnicas como o Prophet e métodos baseados em árvores torna-se vital para o sucesso das iniciativas de energia renovável no contexto da transição energética global.

**Definição da linguagem de programação usada no projeto**

A linguagem de programação usada no projeto aplicado é o Python versão 3.12. O grupo utilizará diversas bibliotecas para manipulação e visualização de dados. A seguir, apresenta-se uma breve descrição de cada uma delas:

* **pandas**: O pandas é uma biblioteca de análise de dados em Python que fornece estruturas de dados eficientes e fáceis de usar, como DataFrames, que permitem a manipulação e análise de dados de forma rápida e eficiente.
* **numpy**: O numpy é uma biblioteca em Python que fornece suporte para arrays multidimensionais e funções matemáticas de alto desempenho. É amplamente utilizado em computação científica e análise numérica de dados.
* **seaborn**: O seaborn é uma biblioteca de visualização de dados baseada no matplotlib que fornece uma interface de alto nível para a criação de gráficos estatísticos atraentes e informativos. Ele é especialmente útil para a criação de gráficos de distribuição, gráficos de regressão e mapas de calor.
* **matplotlib**: O matplotlib é uma biblioteca de plotagem em Python que fornece uma API orientada a objetos para a criação de gráficos estáticos, interativos e animados. Ele é altamente personalizável e oferece suporte a uma ampla variedade de estilos de plotagem.

Provavelmente outras bibliotecas serão utilizadas ao longo do projeto. Desta forma, as atualizações serão efetuadas nas próximas entregas.

**Pipeline e Metodologia**

Para abordar o problema de previsão de energia renovável na Grécia usando a base de dados de geração solar e eólica disponível no repositório do GitHub, propomos um pipeline detalhado para processar os dados e desenvolver modelos preditivos eficazes.

1. Aquisição e Análise Inicial dos Dados

Fonte dos dados: A base de dados foi obtida a partir do repositório fornecido. Ela contém dados históricos de geração solar e eólica.

Formato dos dados: O conjunto de dados deve ser lido em formato CSV ou outro formato adequado. A análise inicial envolve a verificação de variáveis, seu tipo, granularidade temporal, e se a série está agregada por dia, hora ou minuto.

2. Pré-processamento

O pré-processamento é uma etapa fundamental para preparar os dados antes de aplicar os modelos de previsão. As tarefas incluiram:

a) Limpeza de Dados

Remoção de valores ausentes: Verificar se há lacunas nas séries temporais e preenchê-las usando interpolação ou médias móveis.

Remoção de duplicatas: Eliminar registros duplicados que podem distorcer a análise.

Normalização de variáveis: Em alguns casos, normalizar os valores das variáveis pode ajudar a evitar o sobreajuste dos modelos preditivos, especialmente em técnicas de aprendizado de máquina.

b) Tratamento de Outliers

Identificação de outliers: Analisar picos anormais de geração de energia (solar ou eólica) usando boxplots ou a técnica Z-score para detectar valores fora do padrão.

Tratamento: Os outliers podem ser corrigidos com interpolação, valores medianos ou ser removidos, caso sejam erros de medição.

c) Criação de Novas Variáveis (Feature Engineering)

Variáveis derivadas: Calcular médias móveis, diferença entre dias, ou valores sazonais (diurnos, sazonais, ou horários). Isso melhora o poder preditivo do modelo, capturando tendências de longo e curto prazo.

Variáveis climáticas: Se disponível, incluir variáveis climáticas adicionais, como temperatura e nebulosidade, que influenciam diretamente a geração de energia solar.

3. Divisão dos Dados

Divisão entre treinamento e teste: Dividir os dados entre um conjunto de treinamento (70%-80%) e um conjunto de teste (20%-30%). Também é possível usar a validação cruzada (cross-validation) para evitar o sobreajuste.

Estratégia temporal: Como se trata de uma série temporal, a divisão deve respeitar a cronologia, com os dados de teste sendo sempre posteriores aos dados de treinamento.

4. Modelagem Preditiva

A modelagem será feita utilizando dois tipos principais de abordagem:

a) Decomposição com Prophet

O Prophet, um modelo aditivo, será usado para decompor a série temporal em componentes de tendência, sazonalidade, e efeitos residuais. Este método é útil para capturar padrões sazonais e mudanças de tendência.

Modelos aditivos: O Prophet divide o comportamento da série em tendências de longo prazo e padrões de repetição (diurnos, sazonais, etc.). Sua facilidade em ajustar dados sazonais faz dele uma ferramenta robusta para previsões de energia renovável.

b) Métodos Baseados em Árvores

Random Forest (RF) e Gradient Boosting Machines (GBM): Estes métodos são baseados em árvores de decisão e são excelentes para capturar interações não lineares complexas entre variáveis, como variações climáticas e geração de energia.

Feature importance: Esses modelos também são valiosos para entender quais variáveis impactam mais a geração de energia.

5. Validação e Avaliação de Desempenho

A validação do modelo será realizada através de diferentes métricas:

a) Métricas de avaliação

Erro absoluto médio (MAE): Mede a magnitude média dos erros entre as previsões e os valores reais, sem considerar sua direção.

Erro quadrático médio (RMSE): Penaliza mais os grandes erros em relação ao MAE.

Coeficiente de Determinação (R²): Indica a proporção da variabilidade explicada pelo modelo, com valores mais próximos de 1 indicando melhor desempenho.

b) Validação cruzada (cross-validation)

Utilizar validação cruzada em rolling windows para garantir que o modelo tenha robustez e generalize bem os dados não vistos.

6. Implementação e Avaliação Final

Após ajustar os parâmetros e selecionar o melhor modelo, os resultados finais serão aplicados aos dados de teste para avaliação do desempenho preditivo. Comparações serão feitas entre os diferentes modelos (Prophet, Random Forest, GBM) para determinar qual é o mais preciso e robusto.

7. Implantação do Modelo

Automatização do processo: Implementar o modelo final em um ambiente de produção, onde ele possa receber dados novos e gerar previsões de curto prazo continuamente.

Dashboard de visualização: Criar uma interface para que os operadores da rede elétrica possam visualizar as previsões e tomar decisões em tempo real sobre a alocação de energia e ativação de fontes de backup.

8. Aprimoramento Contínuo

Monitoramento e ajuste: À medida que novos dados forem sendo coletados, o modelo poderá ser ajustado continuamente para melhorar a precisão, incorporando dados climáticos atualizados e outras variáveis relevantes.

A metodologia para a análise de séries temporais da geração de energia renovável na Grécia foi ajustada para seguir uma abordagem baseada no pipeline acima descrito, adaptando modelos de predição como *Prophet* e *Tree-based Ensembles* (Random Forest e Gradient Boosting Machines), ao invés de modelos puramente ARIMA. Este pipeline permitirá uma análise mais robusta dos padrões sazonais e de tendências não lineares na geração de energia solar e eólica.

#### 1. Definição do Problema

O problema em questão envolve a necessidade de previsões precisas da geração de energia renovável para melhorar a eficiência, acessibilidade e sustentabilidade, em consonância com o Objetivo de Desenvolvimento Sustentável (ODS) 7 da ONU. A previsão precisa da produção de energia é essencial para assegurar o fornecimento contínuo e eficiente, além de orientar políticas públicas e decisões estratégicas sobre infraestrutura e investimento em energia limpa.

O cenário atual na Grécia, com foco em solar e eólica, traz desafios de intermitência e sazonalidade que afetam a gestão da rede elétrica, justificando a necessidade de previsões mais precisas para otimizar o uso de fontes renováveis e reduzir a dependência de fontes não renováveis.

#### 2. Limpeza de Dados

A limpeza de dados incluirá a remoção de valores ausentes através de interpolação ou médias móveis, tratamento de outliers, como picos anômalos, e a padronização dos dados para a análise subsequente.

O objetivo é garantir que os dados reflitam de maneira fidedigna a variação de geração de energia, evitando que anomalias distorçam os modelos de previsão.

#### 3. Análise Exploratória de Dados (EDA)

A Análise Exploratória de Dados é essencial para compreender as características temporais da geração de energia solar e eólica. Serão utilizados gráficos de linha para a visualização dos dados ao longo do tempo, além de gráficos de autocorrelação (ACF) e de tendências removidas (detrended) para identificar padrões sazonais e de tendências.

Identificar a periodicidade, variações diárias e sazonais permitirão informar a escolha e configuração de modelos preditivos mais adequados.

#### 4. Modelagem de Séries Temporais

Ao invés de ARIMA, será adotado o Prophet para a decomposição aditiva da série temporal, capturando padrões de tendência e sazonalidade. A abordagem do Prophet é especialmente adequada para dados com forte sazonalidade, como a geração de energia solar e eólica.

Modelos baseados em árvores como o Random Forest e o Gradient Boosting Machines (GBM) serão aplicados para modelar relações não lineares entre variáveis (como clima e padrões sazonais) e geração de energia. Esses modelos são capazes de capturar interações complexas e realizar previsões mais precisas.

A criação de variáveis auxiliares, como médias móveis e indicadores climáticos (temperatura, nebulosidade), também será explorada para melhorar o desempenho do modelo.

#### 5. Validação e Avaliação do Modelo

A validação será realizada com a divisão entre conjuntos de treino e teste, respeitando a cronologia dos dados, além da validação cruzada usando *rolling windows* para evitar sobreajuste.

As métricas de avaliação incluem Erro Médio Absoluto (MAE), Erro Quadrático Médio (MSE) e Coeficiente de Determinação (R²), garantindo uma análise robusta da precisão das previsões.

#### 6. Aplicação dos Modelos e Interpretação dos Resultados

Após a seleção do melhor modelo, ele será aplicado para prever a produção futura de energia renovável na Grécia, gerando insights para otimização da alocação de recursos e gestão da infraestrutura energética.

Os resultados serão interpretados à luz das metas do ODS 7, destacando o impacto da previsão na sustentabilidade e eficiência da geração de energia renovável.

#### 7. Relatório e Discussão dos Resultados

Os resultados obtidos serão documentados, analisando as previsões de séries temporais no contexto das metas do ODS 7. A contribuição do estudo para o planejamento e desenvolvimento sustentável da matriz energética será discutida, com sugestões para futuras pesquisas em previsão de energia renovável.

**Descrição da base de dados**

A base de dados utilizada neste trabalho do grupo de Projeto Aplicado 4, para a análise do consumo e geração de energia renovável e não renovável na Grécia foi coletada de fontes confiáveis, como a Hellenic Association of Photovoltaic Companies e a International Renewable Energy Agency (IRENA). Esta base de dados inclui:

Tipo de energia: Identificação da fonte energética, incluindo solar, eólica e outras fontes renováveis.

Consumo de energia: Dados agregados e segmentados por região geográfica e setores econômicos, permitindo uma compreensão clara do consumo energético em diferentes áreas da Grécia.

Geração de energia: Volume de energia gerado por cada tipo de fonte, tanto renovável quanto não renovável, com foco especial na contribuição da energia solar e eólica.

Dados climáticos: Informações sobre temperatura, precipitação e velocidade do vento, que impactam diretamente a geração de energia renovável, especialmente no contexto das fontes solares e eólicas.

Os dados abrangem um período de cinco anos, com coletas diárias e mensais, permitindo uma análise detalhada dos padrões de curto e longo prazo na geração e consumo de energia. A estrutura dos dados é majoritariamente tabular, com registros organizados em uma série temporal. Além disso, também será considerado o uso de bases de dados públicas sobre mudanças climáticas e indicadores econômicos que possam influenciar a oferta e a demanda de energia na Grécia.

**.**

**Análise exploratória da base de dados escolhida**

Conforme descrito na metodologia, apresentam-se abaixo as evidências da análise de dados realizadas sobre a base de dados escolhida pelo grupo.

# Previsão de RES - Pré-processamento de Dados

O conjunto de dados de RES da Grécia combina informações de diferentes fontes. Ele contém dados de geolocalização, clima e produção de energia de RES dos principais contribuintes de energia eólica e solar que representam 20% da energia RES na rede elétrica grega. Os dados climáticos foram coletados de uma API de previsão do tempo e representam a última previsão gerada para cada instalação.

Os dados foram coletados das seguintes fontes:

* Produção de energia RES da [ENTSO-E](https://transparency.entsoe.eu/load-domain/r2/totalLoadR2/show)
* Dados meteorológicos capturados usando a [API Stormglass](https://docs.stormglass.io/#/weather?id=point-request)
* Dados de geolocalização de turbinas eólicas e instalações de PV, para os locais que adquiriram licença de operação, conforme encontrado na [Autoridade Reguladora de Energia (RAE)](https://geo.rae.gr/)

Mais informações sobre licenciamento de energia estão disponíveis em <http://www.opengov.gr/minenv/?p=1031>

from imports import \*

data\_eval = Evaluation()

## 1. Apresentação de Dados

### 1.1 Exploração de Dados

#### 1.1.1 Carregamento de Dados

Carregando os dados combinados das diferentes fontes de dados. O conjunto de dados de energia eólica contém a variável climática dos principais contribuintes de energia eólica e a produção agregada de energia eólica de 2017 a 2020.

In [2]:

# Carregar dados solares

solar\_energy\_path = os.path.join('resources', 'raw\_data', 'solar\_energy\_data.csv')

solar\_energy\_df = pd.read\_csv(solar\_energy\_path, parse\_dates=['time'], date\_parser=lambda col: pd.to\_datetime(col, utc=True),)

solar\_energy\_df.loc[:, 'time'] = solar\_energy\_df.time.dt.tz\_convert('CET')

solar\_energy\_df.sort\_values(['id1', 'time'], inplace=True)

solar\_energy\_df.set\_index('id1', inplace=True)

solar\_energy\_df.head()

Out[2]:

|  | representative\_lat | representative\_lon | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | time | energy\_type | Solar energy (MW) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 11381 | 37.937648 | 23.946727 | 5.54 | 99.00 | 6.60 | 82.90 | 1.37 | 1025.07 | 19.743333 | 320.93 | 5.27 | 2017-01-01 01:00:00+01:00 | solar | 0.0 |
| 11381 | 37.937648 | 23.946727 | 5.53 | 98.17 | 6.13 | 83.37 | 1.22 | 1025.06 | 19.843333 | 323.93 | 4.86 | 2017-01-01 02:00:00+01:00 | solar | 0.0 |
| 11381 | 37.937648 | 23.946727 | 5.51 | 97.33 | 5.67 | 83.83 | 1.08 | 1025.05 | 19.333333 | 326.94 | 4.44 | 2017-01-01 03:00:00+01:00 | solar | 0.0 |
| 11381 | 37.937648 | 23.946727 | 5.50 | 96.50 | 5.20 | 84.30 | 0.93 | 1025.04 | 18.820000 | 329.94 | 4.03 | 2017-01-01 04:00:00+01:00 | solar | 0.0 |
| 11381 | 37.937648 | 23.946727 | 5.60 | 95.67 | 5.27 | 84.20 | 0.78 | 1025.07 | 18.310000 | 332.31 | 4.17 | 2017-01-01 05:00:00+01:00 | solar | 0.0 |

In [3]:

# Carregar dados de energia eólica

wind\_energy\_path = os.path.join('resources', 'raw\_data', 'wind\_energy\_data.csv')

wind\_energy\_df = pd.read\_csv(wind\_energy\_path, parse\_dates=['time'], date\_parser=lambda col: pd.to\_datetime(col, utc=True),)

wind\_energy\_df.loc[:, 'time'] = wind\_energy\_df.time.dt.tz\_convert('CET')

wind\_energy\_df.sort\_values(['id1', 'time'], inplace=True)

wind\_energy\_df.set\_index('id1', inplace=True)

wind\_energy\_df.head()

Out[3]:

|  | representative\_lat | representative\_lon | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | time | energy\_type | Wind energy (MW) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| id1 |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 32947 | 38.774952 | 20.993431 | 4.22 | 79.00 | 3.50 | 63.10 | 0.0 | 1026.40 | 24.010000 | 82.11 | 3.06 | 2017-01-01 01:00:00+01:00 | wind | 381.0 |
| 32947 | 38.774952 | 20.993431 | 4.20 | 65.83 | 3.17 | 63.03 | 0.0 | 1026.18 | 24.110000 | 80.21 | 2.75 | 2017-01-01 02:00:00+01:00 | wind | 391.0 |
| 32947 | 38.774952 | 20.993431 | 4.17 | 52.67 | 2.83 | 62.97 | 0.0 | 1025.97 | 23.670000 | 78.32 | 2.44 | 2017-01-01 03:00:00+01:00 | wind | 355.0 |
| 32947 | 38.774952 | 20.993431 | 4.15 | 39.50 | 2.50 | 62.90 | 0.0 | 1025.76 | 23.233333 | 76.42 | 2.13 | 2017-01-01 04:00:00+01:00 | wind | 360.0 |
| 32947 | 38.774952 | 20.993431 | 4.21 | 26.33 | 2.40 | 62.80 | 0.0 | 1025.77 | 22.793333 | 82.94 | 2.09 | 2017-01-01 05:00:00+01:00 | wind | 330.0 |

In [4]:

config = json.loads(open('config.json', 'r').read())

weather\_features = [var for var in config['weather\_variables'] if var in wind\_energy\_df.columns]

In [5]:

# Verificar a integridade dos dados

for col in wind\_energy\_df.columns:

assert(len(wind\_energy\_df[col].dropna()) == len(wind\_energy\_df.index))

for col in solar\_energy\_df.columns:

assert(len(solar\_energy\_df[col].dropna()) == len(solar\_energy\_df.index))

#### 1.1.2 Mapa dos Principais Contribuintes de RES na Grécia

Focamos nos principais produtores de energia RES que contribuem com 20% da geração total de energia eólica e solar. Apresentamos todas as instalações e destacamos os maiores contribuintes.

In [6]:

# Carregando a configuração do mapa

with open(os.path.join('resources', 'map\_config', 'res\_map\_config.json'), 'r') as res\_map\_config:

res\_map\_config = json.loads(res\_map\_config.read())

In [7]:

# from keplergl import KeplerGl

# res\_map = KeplerGl(height=700, width=400)

# res\_map.add\_data(wind\_energy\_df.append(solar\_energy\_df), 'Top Greek RES Parks')

# res\_map.config = res\_map\_config

# res\_map

In [8]:

# # Saving map style

# with open('map\_config/res\_map\_config.json', 'w') as res\_map\_config:

# json.dump(res\_map.config, res\_map\_config)

#### 1.1.2 Combinação de Dados

Para cada tipo de energia, agregamos as previsões meteorológicas das diferentes localizações para formar um conjunto de dados combinado.

In [9]:

agg\_wind\_energy\_df = wind\_energy\_df.reset\_index().groupby('time').mean().copy()

agg\_wind\_energy\_df.drop(columns=['id1', 'representative\_lat', 'representative\_lon'], inplace=True)

agg\_wind\_energy\_df.tail()

Out[9]:

|  | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | Wind energy (MW) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| time |  |  |  |  |  |  |  |  |  |  |
| 2020-12-31 19:00:00+01:00 | 9.767963 | 21.890000 | 5.894444 | 80.138889 | 0.015926 | 1013.255926 | 24.030000 | 246.156111 | 3.573333 | 1659.0 |
| 2020-12-31 20:00:00+01:00 | 9.420185 | 24.256111 | 5.824815 | 81.385370 | 0.011852 | 1013.739074 | 24.044444 | 245.211481 | 3.403889 | 1516.0 |
| 2020-12-31 21:00:00+01:00 | 9.178704 | 24.375000 | 5.558704 | 81.463148 | 0.025185 | 1014.223519 | 24.065556 | 244.266111 | 3.236296 | 1426.0 |
| 2020-12-31 22:00:00+01:00 | 8.955926 | 26.747407 | 5.263333 | 80.820000 | 0.033333 | 1014.607222 | 24.081667 | 243.321481 | 3.066852 | 1382.0 |
| 2020-12-31 23:00:00+01:00 | 8.728889 | 29.917963 | 5.002593 | 80.800556 | 0.022963 | 1014.706481 | 23.686111 | 243.764074 | 2.862963 | 1416.0 |

In [10]:

agg\_solar\_energy\_df = solar\_energy\_df.reset\_index().groupby('time').mean().copy()

agg\_solar\_energy\_df.drop(columns=['id1', 'representative\_lat', 'representative\_lon'], inplace=True)

agg\_solar\_energy\_df.tail()

Out[10]:

|  | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | Solar energy (MW) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| time |  |  |  |  |  |  |  |  |  |  |
| 2020-12-31 19:00:00+01:00 | 9.114444 | 22.117778 | 7.974444 | 85.508889 | 0.0 | 1014.367778 | 24.13 | 256.385556 | 2.586667 | 0.0 |
| 2020-12-31 20:00:00+01:00 | 8.858889 | 8.531111 | 8.136667 | 82.986667 | 0.0 | 1014.736667 | 24.13 | 260.053333 | 2.441111 | 0.0 |
| 2020-12-31 21:00:00+01:00 | 8.577778 | 3.935556 | 7.850000 | 81.464444 | 0.0 | 1015.200000 | 24.14 | 263.715556 | 2.303333 | 0.0 |
| 2020-12-31 22:00:00+01:00 | 8.315556 | 2.017778 | 7.344444 | 79.937778 | 0.0 | 1015.472222 | 24.14 | 267.383333 | 2.157778 | 0.0 |
| 2020-12-31 23:00:00+01:00 | 8.035556 | 3.044444 | 6.901111 | 75.550000 | 0.0 | 1015.524444 | 24.14 | 273.384444 | 2.074444 | 0.0 |

### 1.2 Engenharia de Features

Recursos temporais são adicionados e divididos em cíclicos e discretos. Os recursos cíclicos, como hora, mês, dia da semana, dia do mês e dia do ano, são traduzidos para coordenadas polares, enquanto os recursos não cíclicos, como ano e estação, são adicionados como one-hot encoding. Durante a fase de engenharia de recursos, o conjunto de dados passa pelas seguintes etapas:

* cálculo de recursos temporais
* codificação polar e one-hot dos recursos temporais
* adição da produção de energia do último dia como recurso
* normalização da coluna alvo

In [11]:

# Conjuntos de recursos

temporal\_features = ['hour', 'day', 'weekday\_index', 'dayofyear', 'month', 'year', 'season', 'timeofday']

cyclical\_features = ['windDirection', 'hour', 'month', 'day', 'dayofyear', 'weekday\_index']

weather\_features = list(set(weather\_features) - set(['windDirection']))

one\_hot\_features = ['year', 'season', 'timeofday']

# Cálculo do horário do dia

def get\_time\_of\_day(instance):

instance = pd.to\_datetime(instance)

location = LocationInfo("Athens", "Greece").observer

phases = sun(location, date=instance)

phases = {k:pd.to\_datetime(phases[k]).tz\_convert('CET') for k in phases.keys()}

if phases['dawn'] < instance <= phases['sunrise']:

return 'sunrise'

elif phases['sunrise'] < instance <= phases['noon']:

return 'morning'

elif phases['noon'] < instance <= phases['sunset']:

return 'noon'

elif phases['sunset'] < instance <= phases['dusk']:

return 'sunset'

else:

return 'night'

# Cálculo da estação

def season\_calc(month):

if month in [6, 7, 8, 9, 10]:

return "summer"

else:

return "winter"

def add\_temporal\_features(df):

df['hour'] = df.index.hour

df['datetime'] = df.index

df['month'] = df.index.month

df['year'] = df.index.year

df['day'] = df.index.day

df['dayofyear'] = df.index.dayofyear

df['weekday\_index'] = df.index.dayofweek

df['season'] = df['month'].apply(season\_calc)

df['timeofday'] = df['datetime'].apply(get\_time\_of\_day)

return df.drop(columns=['datetime'])

agg\_wind\_energy\_df = agg\_wind\_energy\_df.pipe(add\_temporal\_features)

print(agg\_wind\_energy\_df.columns)

agg\_solar\_energy\_df = agg\_solar\_energy\_df.pipe(add\_temporal\_features)

print(agg\_solar\_energy\_df.columns)

Index(['airTemperature', 'cloudCover', 'gust', 'humidity', 'precipitation',

'pressure', 'visibility', 'windDirection', 'windSpeed',

'Wind energy (MW)', 'hour', 'month', 'year', 'day', 'dayofyear',

'weekday\_index', 'season', 'timeofday'],

dtype='object')

Index(['airTemperature', 'cloudCover', 'gust', 'humidity', 'precipitation',

'pressure', 'visibility', 'windDirection', 'windSpeed',

'Solar energy (MW)', 'hour', 'month', 'year', 'day', 'dayofyear',

'weekday\_index', 'season', 'timeofday'],

dtype='object')

Ao explorar os padrões temporais da energia solar, torna-se evidente que há uma forte sazonalidade durante o dia e nos meses de verão. Além disso, há uma tendência de aumento na produção de energia solar ao longo dos anos e do inverno para o verão.

In [12]:

fig, ax = plt.subplots(len(temporal\_features), 1, figsize = (8, 15))

for i,feature in enumerate(temporal\_features):

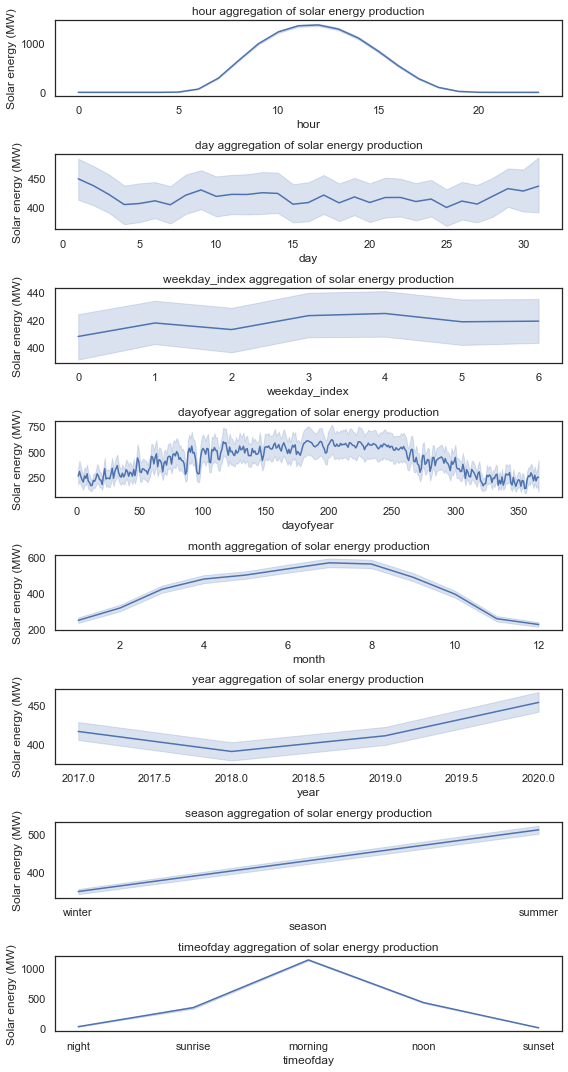
\_ = sns.lineplot(x=feature,

y='Solar energy (MW)',

data=agg\_solar\_energy\_df,

ax=ax[i]).set\_title('%s aggregation of solar energy production' % feature)

plt.tight\_layout()



Explorando os padrões temporais da produção de energia eólica, torna-se evidente que não há padrões sazonais, o que evidencia a natureza intermitente dos elementos do vento. Além disso, há uma tendência de aumento na produção de energia eólica ao longo dos anos e uma diminuição do inverno para o verão.

In [13]:

fig, ax = plt.subplots(len(temporal\_features), 1, figsize = (8, 15))

for i,feature in enumerate(temporal\_features):

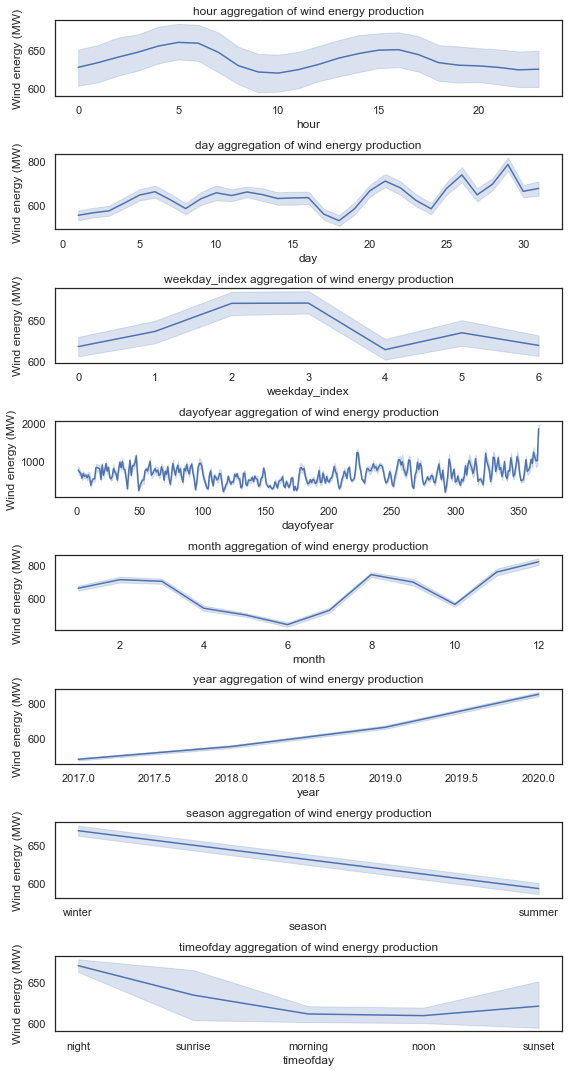
\_ = sns.lineplot(x=feature,

y='Wind energy (MW)',

data=agg\_wind\_energy\_df,

ax=ax[i]).set\_title('%s aggregation of wind energy production' % feature)

plt.tight\_layout()



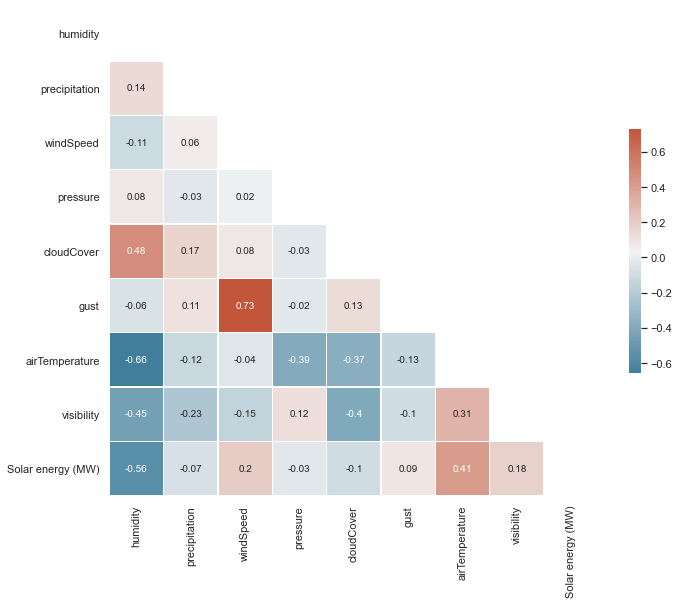
In [14]:

data\_eval.plot\_corr\_heatmap(agg\_solar\_energy\_df, weather\_features + ['Solar energy (MW)'])

agg\_solar\_energy\_df.tail()

Out[14]:

|  | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | Solar energy (MW) | hour | month | year | day | dayofyear | weekday\_index | season | timeofday |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| time |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2020-12-31 19:00:00+01:00 | 9.114444 | 22.117778 | 7.974444 | 85.508889 | 0.0 | 1014.367778 | 24.13 | 256.385556 | 2.586667 | 0.0 | 19 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 20:00:00+01:00 | 8.858889 | 8.531111 | 8.136667 | 82.986667 | 0.0 | 1014.736667 | 24.13 | 260.053333 | 2.441111 | 0.0 | 20 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 21:00:00+01:00 | 8.577778 | 3.935556 | 7.850000 | 81.464444 | 0.0 | 1015.200000 | 24.14 | 263.715556 | 2.303333 | 0.0 | 21 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 22:00:00+01:00 | 8.315556 | 2.017778 | 7.344444 | 79.937778 | 0.0 | 1015.472222 | 24.14 | 267.383333 | 2.157778 | 0.0 | 22 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 23:00:00+01:00 | 8.035556 | 3.044444 | 6.901111 | 75.550000 | 0.0 | 1015.524444 | 24.14 | 273.384444 | 2.074444 | 0.0 | 23 | 12 | 2020 | 31 | 366 | 3 | winter | night |



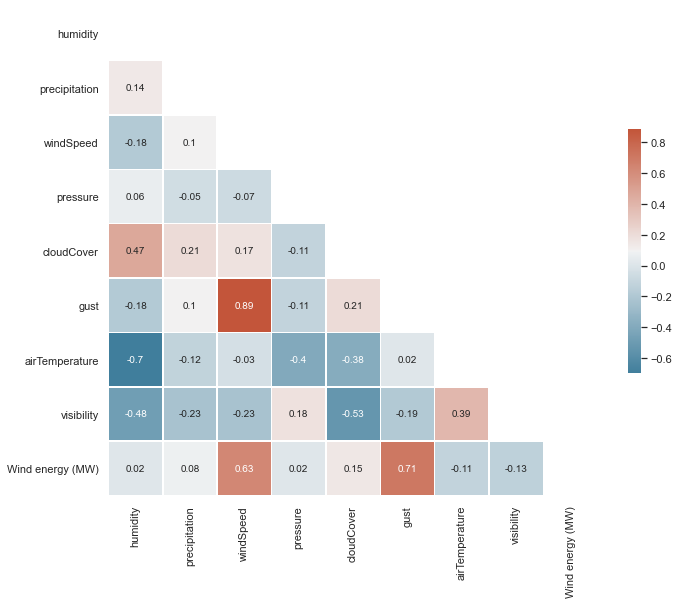
In [15]:

data\_eval.plot\_corr\_heatmap(agg\_wind\_energy\_df, weather\_features + ['Wind energy (MW)'])

agg\_wind\_energy\_df.tail()

Out[15]:

|  | airTemperature | cloudCover | gust | humidity | precipitation | pressure | visibility | windDirection | windSpeed | Wind energy (MW) | hour | month | year | day | dayofyear | weekday\_index | season | timeofday |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| time |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| 2020-12-31 19:00:00+01:00 | 9.767963 | 21.890000 | 5.894444 | 80.138889 | 0.015926 | 1013.255926 | 24.030000 | 246.156111 | 3.573333 | 1659.0 | 19 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 20:00:00+01:00 | 9.420185 | 24.256111 | 5.824815 | 81.385370 | 0.011852 | 1013.739074 | 24.044444 | 245.211481 | 3.403889 | 1516.0 | 20 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 21:00:00+01:00 | 9.178704 | 24.375000 | 5.558704 | 81.463148 | 0.025185 | 1014.223519 | 24.065556 | 244.266111 | 3.236296 | 1426.0 | 21 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 22:00:00+01:00 | 8.955926 | 26.747407 | 5.263333 | 80.820000 | 0.033333 | 1014.607222 | 24.081667 | 243.321481 | 3.066852 | 1382.0 | 22 | 12 | 2020 | 31 | 366 | 3 | winter | night |
| 2020-12-31 23:00:00+01:00 | 8.728889 | 29.917963 | 5.002593 | 80.800556 | 0.022963 | 1014.706481 | 23.686111 | 243.764074 | 2.862963 | 1416.0 | 23 | 12 | 2020 | 31 | 366 | 3 | winter | night |



In [16]:

\_ = plt.figure(figsize=(12, 8))

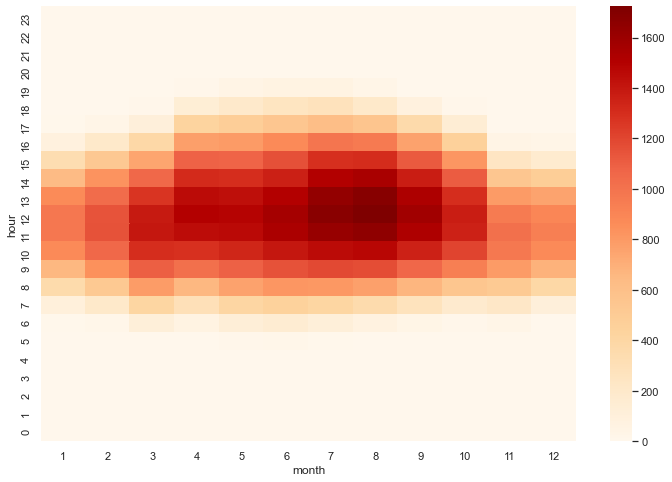
ax = sns.heatmap(agg\_solar\_energy\_df.pivot\_table(values='Solar energy (MW)',

index='hour',

columns = 'month',

aggfunc = 'mean').sort\_index(ascending = False),

cmap='OrRd')



In [17]:

# sns.set(font\_scale = 2.5)

\_ = plt.figure(figsize=(12, 8))

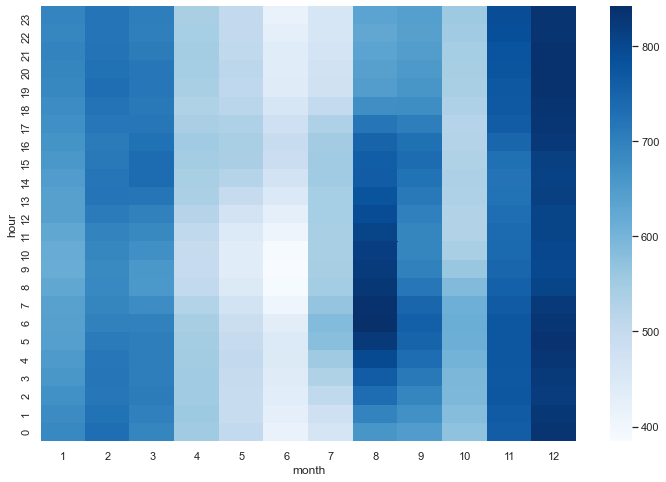
ax = sns.heatmap(agg\_wind\_energy\_df.pivot\_table(values='Wind energy (MW)',

index='hour',

columns = 'month',

aggfunc = 'mean').sort\_index(ascending = False),

cmap='Blues')



#### 1.2.1 Recursos Cíclicos (polares)

Os recursos temporais são, de fato, cíclicos por natureza, então, em vez de aplicá-los como one hot encoding, tentamos uma representação com coordenadas polares. Para cada recurso, representamos seu valor (x) como:xsin=sin⁡(2∗π∗xmax(x))xcos=cos⁡(2∗π∗xmax(x))

In [18]:

for col in cyclical\_features:

# sol

agg\_solar\_energy\_df['sin\_%s' % col] = agg\_solar\_energy\_df[col].apply(lambda x: np.sin((2 \* np.pi \* x)/agg\_solar\_energy\_df[col].max()))

agg\_solar\_energy\_df['cos\_%s' % col] = agg\_solar\_energy\_df[col].apply(lambda x: np.cos((2 \* np.pi \* x)/agg\_solar\_energy\_df[col].max()))

# vento

agg\_wind\_energy\_df['sin\_%s' % col] = agg\_wind\_energy\_df[col].apply(lambda x: np.sin((2 \* np.pi \* x)/agg\_wind\_energy\_df[col].max()))

agg\_wind\_energy\_df['cos\_%s' % col] = agg\_wind\_energy\_df[col].apply(lambda x: np.cos((2 \* np.pi \* x)/agg\_wind\_energy\_df[col].max()))

#### 1.2.2 Recursos codificados em one-hot

A codificação one-hot é escolhida para alguns dos recursos, como **estação** e **ano**.

In [19]:

for col in one\_hot\_features:

agg\_solar\_energy\_df[col] = agg\_solar\_energy\_df[col].astype('category')

agg\_wind\_energy\_df[col] = agg\_wind\_energy\_df[col].astype('category')

# Preparing dummy columns

agg\_wind\_energy\_df = pd.get\_dummies(agg\_wind\_energy\_df)

agg\_solar\_energy\_df = pd.get\_dummies(agg\_solar\_energy\_df)

#### 1.2.3 Produção do último dia como recurso

Adicionamos a produção de energia da semana anterior como um recurso. Juntamente com as variáveis climáticas, de acordo com a correlação exibida abaixo.

Calcule os recursos da semana passada:

* geração de energia horária da semana passada
* variáveis climáticas horárias da semana passada
* estatísticas móveis da geração de energia de t-24h a t-72h a cada 3 horas

In [20]:

# Obter características climáticas dos últimos 3 dias

for h in range(1, 72, 2):

agg\_solar\_energy\_df['t-%sh temp' % h] = agg\_solar\_energy\_df['airTemperature'].shift(freq=timedelta(hours=h))

agg\_solar\_energy\_df['t-%sh humidity' % h] = agg\_solar\_energy\_df['humidity'].shift(freq=timedelta(hours=h))

agg\_wind\_energy\_df['t-%sh wind' % h] = agg\_wind\_energy\_df['windSpeed'].shift(freq=timedelta(hours=h))

agg\_wind\_energy\_df['t-%sh gust' % h] = agg\_wind\_energy\_df['gust'].shift(freq=timedelta(hours=h))

In [21]:

# Obter características climáticas um dia à frente

for h in range(1, 24, 2):

agg\_solar\_energy\_df['t+%sh temp' % h] = agg\_solar\_energy\_df['airTemperature'].shift(freq=timedelta(hours=-h))

agg\_solar\_energy\_df['t+%sh humidity' % h] = agg\_solar\_energy\_df['humidity'].shift(freq=timedelta(hours=-h))

agg\_wind\_energy\_df['t+%sh wind' % h] = agg\_wind\_energy\_df['windSpeed'].shift(freq=timedelta(hours=-h))

agg\_wind\_energy\_df['t+%sh gust' % h] = agg\_wind\_energy\_df['gust'].shift(freq=timedelta(hours=-h))

In [22]:

# Obter dados de energia de 2 dias (48h) antes do dia alvo até 1 semana (168h)

for h in range(48, 168):

agg\_solar\_energy\_df['t-%sh energy' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].shift(freq=timedelta(hours=h))

agg\_wind\_energy\_df['t-%sh energy' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].shift(freq=timedelta(hours=h))

In [23]:

# Obter estatísticas de energia de 48 a 96 horas antes do dia alvo

for h in range(48, 96, 6):

agg\_solar\_energy\_df.loc[:, 'rolling\_max\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).max()

agg\_solar\_energy\_df.loc[:, 'rolling\_min\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).min()

agg\_solar\_energy\_df.loc[:, 'rolling\_mean\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).mean()

agg\_solar\_energy\_df.loc[:, 'rolling\_var\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).var()

agg\_solar\_energy\_df.loc[:, 'rolling\_std\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).std()

agg\_solar\_energy\_df.loc[:, 'rolling\_skew\_energy\_%sh\_win' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].rolling(window='%sH' % h).skew()

agg\_wind\_energy\_df.loc[:, 'rolling\_max\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).max()

agg\_wind\_energy\_df.loc[:, 'rolling\_min\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).min()

agg\_wind\_energy\_df.loc[:, 'rolling\_mean\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).mean()

agg\_wind\_energy\_df.loc[:, 'rolling\_var\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).var()

agg\_wind\_energy\_df.loc[:, 'rolling\_std\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).std()

agg\_wind\_energy\_df.loc[:, 'rolling\_skew\_energy\_%sh\_win' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].rolling(window='%sH' % h).skew()

In [24]:

# Obter a produção de energia da próxima semana

for h in range(168):

agg\_solar\_energy\_df['t+%sh energy' % h] = agg\_solar\_energy\_df['Solar energy (MW)'].shift(freq=timedelta(hours=-h))

agg\_wind\_energy\_df['t+%sh energy' % h] = agg\_wind\_energy\_df['Wind energy (MW)'].shift(freq=timedelta(hours=-h))

# Remover a coluna de produção de energia solar original

agg\_solar\_energy\_df.drop(columns='Solar energy (MW)', inplace=True)

# Remover a coluna de produção de energia eólica original

agg\_wind\_energy\_df.drop(columns='Wind energy (MW)', inplace=True)

#### Escalar as features de energia e clima com MinMaxScaler e remover características redundantes

In [25]:

# excluir linhas em excesso

agg\_solar\_energy\_df.dropna(inplace=True)

agg\_wind\_energy\_df.dropna(inplace=True)

print('Data size before feature selection', agg\_solar\_energy\_df.shape, agg\_wind\_energy\_df.shape)

Data size before feature selection (34729, 469) (34729, 469)

#### Escalando os dados

In [26]:

selected\_wind = weather\_features + agg\_wind\_energy\_df.filter(like='energy').columns.tolist()

selected\_solar = weather\_features + agg\_solar\_energy\_df.filter(like='energy').columns.tolist()

# escalando

agg\_wind\_energy\_df.loc[:, selected\_wind] = MinMaxScaler().fit\_transform(agg\_wind\_energy\_df[selected\_wind].values)

agg\_solar\_energy\_df.loc[:, selected\_solar] = MinMaxScaler().fit\_transform(agg\_solar\_energy\_df[selected\_solar].values)

# excluir features em excesso

agg\_wind\_energy\_df.drop(columns=cyclical\_features, inplace=True)

agg\_solar\_energy\_df.drop(columns=cyclical\_features, inplace=True)

**Features Selecionadas**

In [27]:

print(agg\_wind\_energy\_df.shape)

print(agg\_solar\_energy\_df.shape)

(34729, 463)

(34729, 463)

## 2. Divisão de Dados

Nós dividimos os dados para os diferentes modelos que serão treinados posteriormente:

* **Divisão de uma hora à frente**: modelo de previsão para uma hora à frente.
* **Divisão de um dia à frente**: previsão para um dia à frente.
* **Divisão de uma semana à frente**: previsão para uma semana à frente.

### 2.1 Previsão de 1 passo à frente

Divida o conjunto de dados de 4 anos em um conjunto de treinamento de 3 anos e um conjunto de teste de 1 ano. Os anos de 2017 a 2019 serão usados para treinamento, e o último ano, 2020, será utilizado para previsões.

In [33]:

def select\_features(data\_dest\_dir, filename\_train, filename\_test, train\_data, test\_data, selection\_model, target\_variable):

# criar diretorio de exportação

if not os.path.exists(data\_dest\_dir):

os.makedirs(data\_dest\_dir)

start = datetime.now()

print('Feature selection...')

final\_features = data\_eval.final\_feature\_selection(model,

train\_data.drop(columns=target\_variable),

train\_data.loc[:, target\_variable])

final\_features += target\_variable

print('Finished feature selection in %s' % str(datetime.now()-start))

print('Saving wind feature selection')

train\_data[final\_features].to\_csv(os.path.join(data\_dest\_dir, filename\_train))

test\_data[final\_features].to\_csv(os.path.join(data\_dest\_dir, filename\_test))

print('Data: \nFeatures: %s\ntrain: %s\ntest: %s\n' % (train\_data[final\_features].columns,

train\_data[final\_features].shape,

test\_data[final\_features].shape))

return train\_data[final\_features], test\_data[final\_features]

Escolher as características de rolagem e atraso das últimas semanas de 48h a 96h atrás

In [34]:

# Definir variável alvo e ignorar as outras características de look ahead

target\_variable = agg\_wind\_energy\_df.filter(regex='t\+.+ energy').columns.tolist()[:1]

all\_target\_variables = agg\_wind\_energy\_df.filter(regex='t\+.+ energy').columns.tolist()

redundant\_target\_variables = [var for var in all\_target\_variables if var not in target\_variable]

# Filtrar características em excesso

all\_extra\_features = agg\_wind\_energy\_df.filter(regex='t\-|rolling').columns.tolist()

lag\_weather = ['t-%sh %s' % (h,w) for h in range(24) for w in ['gust', 'wind']]

lag\_energy = ['t-%sh energy' % h for h in range(48, 96)]

rolling\_features = agg\_wind\_energy\_df.filter(regex='rolling.+').columns.tolist()

redundant\_variables = [var for var in all\_extra\_features if var not in lag\_weather + lag\_energy + rolling\_features]

# Criar diretório de exportação

data\_dest\_dir = os.path.join('resources', 'forecasting\_data', '1-step-ahead')

if not os.path.exists(data\_dest\_dir):

os.makedirs(data\_dest\_dir)

# Divisão de treino

train\_data = agg\_wind\_energy\_df.loc['2017':'2019'].copy()

train\_data.drop(columns=redundant\_target\_variables + redundant\_variables, inplace=True)

# Divisão de teste

test\_data = agg\_wind\_energy\_df.loc['2020'].copy()

test\_data.drop(columns=redundant\_target\_variables + redundant\_variables, inplace=True)

model = Ridge(normalize=True, tol=1e-5)

# model = xgb.XGBRegressor()

train\_data, test\_data = select\_features(data\_dest\_dir,

'wind\_train\_hour.csv',

'wind\_test\_hour.csv',

train\_data,

test\_data,

model,

target\_variable)

###################################3/

## Conjunto de dados de energia solar

# Definir variável alvo e ignorar as outras características de look ahead

target\_variable = agg\_solar\_energy\_df.filter(regex='t\+.+ energy').columns.tolist()[:1]

all\_target\_variables = agg\_solar\_energy\_df.filter(regex='t\+.+ energy').columns.tolist()

redundant\_target\_variables = [var for var in all\_target\_variables if var not in target\_variable]

# Filtrar características em excesso

all\_extra\_features = agg\_solar\_energy\_df.filter(regex='t\-|rolling').columns.tolist()

lag\_weather = ['t-%sh %s' % (h,w) for h in range(24) for w in ['temp', 'humidity']]

lag\_energy = ['t-%sh energy' % h for h in range(48, 96)]

rolling\_features = agg\_solar\_energy\_df.filter(regex='rolling.+').columns.tolist()

redundant\_variables = [var for var in all\_extra\_features if var not in lag\_weather + lag\_energy + rolling\_features]

# Divisão de treino

train\_data = agg\_solar\_energy\_df.loc['2017':'2019'].copy()

train\_data.drop(columns=redundant\_target\_variables + redundant\_variables, inplace=True)

# Divisão de teste

test\_data = agg\_solar\_energy\_df.loc['2020'].copy()

test\_data.drop(columns=redundant\_target\_variables + redundant\_variables, inplace=True)

model = Ridge(normalize=True, tol=1e-5)

# model = xgb.XGBRegressor()

train\_data, test\_data = select\_features(data\_dest\_dir,

'solar\_train\_hour.csv',

'solar\_test\_hour.csv',

train\_data,

test\_data,

model,

target\_variable)

Feature selection...

Finished feature selection in 0:00:25.028088

Saving wind feature selection

Data:

Features: Index(['airTemperature', 'cloudCover', 'gust', 'humidity', 'precipitation',

'pressure', 'visibility', 'windSpeed', 'sin\_windDirection',

'cos\_windDirection',

...

'rolling\_var\_energy\_84h\_win', 'rolling\_std\_energy\_84h\_win',

'rolling\_skew\_energy\_84h\_win', 'rolling\_max\_energy\_90h\_win',

'rolling\_min\_energy\_90h\_win', 'rolling\_mean\_energy\_90h\_win',

'rolling\_var\_energy\_90h\_win', 'rolling\_std\_energy\_90h\_win',

'rolling\_skew\_energy\_90h\_win', 't+0h energy'],

dtype='object', length=168)

train: (26112, 168)

test: (8617, 168)

Feature selection...

Finished feature selection in 0:00:23.179036

Saving wind feature selection

Data:

Features: Index(['airTemperature', 'cloudCover', 'gust', 'humidity', 'precipitation',

'pressure', 'visibility', 'windSpeed', 'sin\_windDirection',

'cos\_windDirection',

...

'rolling\_mean\_energy\_84h\_win', 'rolling\_var\_energy\_84h\_win',

'rolling\_skew\_energy\_84h\_win', 'rolling\_max\_energy\_90h\_win',

'rolling\_min\_energy\_90h\_win', 'rolling\_mean\_energy\_90h\_win',

'rolling\_var\_energy\_90h\_win', 'rolling\_std\_energy\_90h\_win',

'rolling\_skew\_energy\_90h\_win', 't+0h energy'],

dtype='object', length=150)

train: (26112, 150)

test: (8617, 150)

///

**Tratamento da base de dados, Preparação e treinamento**

Em relação ao tratamento da base de dados, expõem-se a seguir as evidências da metodologia previamente informada.

Este capítulo será apresentado nas próximas etapas

**Resultados Esperados**

Espera-se desenvolver e implementar modelos preditivos de séries temporais para analisar e prever padrões de geração e consumo de energia renovável na Grécia. A intenção é reduzir a dependência de fontes de energia não renováveis, promovendo um sistema energético mais eficiente e sustentável. O projeto pretende alcançar os seguintes resultados específicos:

Identificação de Padrões: Analisar dados históricos da geração e consumo de energia renovável, identificando tendências e padrões cíclicos específicos da região, como sazonalidades influenciadas pelas condições climáticas.

Modelagem Preditiva: Desenvolver modelos preditivos utilizando técnicas de séries temporais, adaptando-os às características dos dados de energia solar e eólica da Grécia.

Simulação e Otimização: Simular cenários futuros de oferta e demanda energética na Grécia, otimizando o uso de fontes renováveis e minimizando o desperdício de energia, contribuindo para a sustentabilidade e a eficiência do sistema energético local.

**Conclusões e Trabalhos Futuros**

Este capítulo será apresentado em etapas futuras

**Referências Bibliográficas**

MORETTIN, P. A., & TOLOI, C. M. C. "Análise de Séries Temporais." Edusp, São Paulo, 2006. Disponível online em <https://doceru.com/doc/ncx81cv>. Acesso em: 29 ago. 2024

HYNDMAN, R. J. *Box-Jenkins modeling*. Disponível em: <https://robjhyndman.com/papers/BoxJenkins.pdf>. Acesso em: 29 ago. 2024.

Organização das Nações Unidas. Objetivos de Desenvolvimento Sustentável. Disponível em: <https://brasil.un.org/pt-br/sdgs/7>. Acesso em: 30 ago. 2024.

ALMEIDA, M. P.; SILVA, J. C. *Modelagem de séries temporais para previsão de demanda energética*. Revista Brasileira de Energia, v. 25, n. 3, p. 45-62, 2023.

MENA, R., GIL, E., TORO, N., et al. *Short-term wind power forecast based on ARIMA model in a wind farm in Antofagasta, Chile*. Renewable Energy, v. 123, p. 119-127, 2018. DOI: 10.1016/j.renene.2018.02.043

HUANG, Y. et al. Energy efficient streaming time series classification with attentive power iteration. *Proceedings of the AAAI Conference on Artificial Intelligence*, v. 38, n. 4, p. 297-305, 2024. Disponível em: [www.aaai.org](https://aaai.org). Acesso em: 22 set. 2024.

SHAHZAD, Sulman; ZAHEER, Asad; et al. Short-Term Load Forecasting Models: A Review of Challenges, Progress, and the Road Ahead. Energies, v. 16, n. 10, p. 4060, 2023. DOI: [10.3390/en16104060](https://doi.org/10.3390/en16104060). Disponível em: <https://www.mdpi.com/1996-1073/16/10/4060>. Acesso em: 22 set. 2024.

Benti, N. E., Chaka, M. D., & Semie, A. G. (2023). Forecasting Renewable Energy Generation with Machine Learning and Deep Learning: Current Advances and Future Prospects. *Sustainability*, 15(9), 7087. <https://doi.org/10.3390/su15097087>. Disponível em: [www. https://www.mdpi.com/2071-1050/15/9/7087](https://www.elsevier.com). Acesso em: 29 set. 2024.

PACELLA, Massimo; PAPA, Antonio; PAPADIA, Gabriele. On Integrating Time-Series Modeling with Long Short-Term Memory and Bayesian Optimization: A Comparative Analysis for Photovoltaic Power Forecasting. Applied Sciences, v. 14, n. 8, p. 3217, 2024. DOI: 10.3390/app14083217. Disponível em: https://www.mdpi.com/2076-3417/14/8/3217. Acesso em: 24 set. 2024.

Padilha, G. A. G., Ko, J., Jung, J. J., & de Mattos Neto, P. S. G. (2022). Transformer-Based Hybrid Forecasting Model for Multivariate Renewable Energy. *Applied Sciences*, 12(21), 10985. <https://doi.org/10.3390/app122110985> Disponível em: https://www.mdpi.com/2076-3417/12/21/10985. Acesso em: 29 set. 2024

Hellenic Association of Photovoltaic Companies. (2023). Solar Photovoltaic Capacity in Greece. Disponível em: <https://helapco.gr>. Acesso em: 29 set. 2024

Agência Nacional de Energia Elétrica (ANEEL). (2023). Relatório Anual de Capacidade Instalada de Energias Renováveis no Brasil. Disponível em: <https://www.aneel.gov.br>. Acesso em: 30 ago. 2024

International Renewable Energy Agency (IRENA). (2023). Renewable Energy Technologies and Carbon Emission Reductions. Disponível em: <https://www.irena.org>. Acesso em: 29 set. 2024

Zafirakis, D., & Tsoutsos, T. (2023). Short-term Renewable Energy Forecasting in Greece using Prophet Decomposition and Tree-based Ensembles. International Journal of Renewable Energy Research, 13(1). Disponível em: <https://www.ijrer.org>. Acesso em: 29 set. 2024

**Cronograma de atividades**

|  |  |  |
| --- | --- | --- |
| **PERÍODO** | **DESCRIÇÃO DA ATIVIDADE** | **STATUS** |
| De 01-08 a 20-08 | Definição do grupo | Concluído |
| De 21-08 a 26-08 | Definição do tema | Concluído |
| De 27-08 a 05-09 | Desenvolvimento da primeira entrega | Concluído |
| De 04-09 a 05-09 | Criação do GITHUB | Concluído |
| De 04-09 a 05-09 | Revisão e aprovação dos itens de capa, sumário, introdução, coleta de dados, resultados esperados, cronograma e demais requerimentos sobre o trabalho a ser desenvolvido | Concluído |
| De 05-09 a 05-09 | Subida dos dados no repositório | Concluído |
| Dia 06-09 | Entrega do Aplicando Conhecimento 1 | Concluído |
| De 07-09 a 26-09 | Desenvolvimento da segunda entrega | Concluído |
| De 07-09 a 20-09 | Desdobramento teórico da proposta analítica baseado nos referenciais lidos e estudados | Concluído |
| De 20-09 a 24-09 | Construção do pipeline da solução proposta | Concluído |
| De 25-09 a 29-09 | Atualização do repositório, projeto e documentos | Concluído |
| Dia 30-09 | Entrega do Aplicando Conhecimento 2 | Concluído |
| De 01-10 a 23-10 | Desenvolvimento da terceira entrega | Concluído |
| De 01-10 a 05-10 | Análise exploratória e Pré-processamento dos dados (limpeza, outliers, etc) | Concluído |
| De 06-10 a 16-10 | Construção da codificação do modelo base conforme método descrito no trabalho.  Esboço do Storytelling. | Concluído |
| De 16-10 a 23-10 | Atualização do pipeline e Cronograma. Avaliação de eventuais necessidades de ajustes | Concluído |
| De 24-10 a 27-10 | Atualização do repositório no Github, projeto e documentos (revisão de tudo para entrega) | Concluído |
| Dia 28-10 | Entrega do Aplicando Conhecimento 3 | Concluído |
| De 29-10 a 11-11 | Desenvolvimento da quarta entrega | Em desenvolvimento |
| De 01-11 a 05-11 | Divisão para speakers baseado no notebook do projeto e artigo do projeto para a construção dos vídeos solicitados | Em desenvolvimento |
| De 06-11 a 07-11 | Gravação do Vídeo de Apresentação do Projeto e Vídeo de Apresentação da Solução Técnica e postagem no YouTube com link no Github ou outro onde o prof. orientar | Em desenvolvimento |
| De 07-11 a 11-11 | Subida do projeto final no GitHub com todos os artefatos e documentos do projeto, bem como códigos do projeto incluindo notebooks, readme de apresentação, etc. | Em desenvolvimento |
| De 12-11 a 17-11 | Revisão e correção da gravação de todo o conteúdo postado no Github e You Tube, se necessário | Em desenvolvimento |
| Dia 18-11 | Entrega do Aplicando Conhecimento 4 | Em desenvolvimento |

GITHUB e YouTube

Para aqueles interessados em explorar mais a fundo os aspectos técnicos e metodológicos do projeto aqui apresentado, serão disponibilizados todos os metadados coletados, bem como o desenvolvimento do algoritmo e a aplicação do projeto, em nosso repositório no GitHub. O repositório pode ser acessado através do seguinte link: https://github.com/BelBatanete/Projeto-Aplicado-4. Este repositório oferece uma visão detalhada de cada etapa do desenvolvimento, desde a coleta e pré-processamento dos dados até a implementação e resultados finais.

Além disso, um vídeo explicativo será criado para facilitar a compreensão do trabalho realizado. Este vídeo estará disponível no YouTube e pode ser acessado pelo link que será informado futuramente. No vídeo, serão abordadas as técnicas utilizadas e os resultados obtidos.