Previsão da Produção de Biodiesel com Rede Neural do Tipo LSTM Bidirecional

Aluno: André de Oliveira Salles

Orientador: Felipe Borges.

Trabalho apresentado ao curso BI MASTER como pré-requisito para conclusão de curso e obtenção de crédito na disciplina "Projetos de Sistemas Inteligentes de Apoio à Decisão".

- Link para o código.
- Link para fonte de arquivos

Resumo

O projeto teve como objetivo utilizar uma rede neural do tipo LSTM bidirecional para prever futuros valores de produção de biodiesel para todo o Brasil, a partir dos seus dados históricos. Os dados foram obtidos na plataforma de dados abertos do governo federal e todo o desenvolvimento em Python foi realizado no ambiente virtual do Google Colab. A previsão foi realizada a partir de uma combinação de diversas camadas de uma rede lstm com uma rede bidirecional. O resultado foi capaz de captar a tendência das séries de cada região, porém as nuâncias da rede não foram modeladas corretamente.

1. Introdução

Assim como diversos dados fornecidos pelo governo federal, a informação da produção de biodiesel está aberta para o acesso de qualquer pessoa no site de dados abertos. Até o desenvolvimento do trabalho, os dados são mensais fornecidos eram do período de 2005 até 2020 e podem ser classificados de acordo com a região do Brasil, contendo, além disso, informações informações sobre as unidades da federação e produtor.

Para realizar a predição de um valor futuro da série temporal, foram utilizadas redes neurais para cada região do Brasil. Devido a sua capacidade de lembrar dos parâmetros de entrada seguindo uma sequência de dados, o tipo de rede utilizada foi a LSTM desenvolvida para o Tensorflow. Além disso, com o objetivo de tentar melhorar o treinamento da rede o modelo de rede bidirecional foi acoplado a estrutura da rede LSTM.

O projeto foi totalmente desenvolvido no ambiente em virtual da Google, o Google Colab, devido a praticidade para fazer o upload dos dados históricos da produção de biodiesel e da praticidade em instalar as bibliotecas necessárias. Todas esão listadas abaixo:

- numpy as np
- matplotlib.pyplot as plt
- pandas as pd
- os
- datetime import datetime
- keras.models import Sequential
- keras.layers import LSTM
- keras.layers import Dense
- keras.layers import Bidirectional
- keras.layers import Dropout
- datetime import datetime
- sklearn.preprocessing import MinMaxScaler
- sklearn.model_selection import train_test_split
- tensorflow.python.keras.preprocessing.sequence import TimeseriesGenerator
- keras.callbacks import EarlyStopping, ReduceLROnPlateau, ModelCheckpoint
- math
- sklearn.metrics import mean_squared_error
- tensorflow import keras

2. Modelagem

O primeiro passo do trabalho consistiu em fazer o upload do arquivo '.csv' para a pasta de armazenamento do Google Colab. Uma vez com os dados da produção de biodiesel no ambiente, o comando 'read_me' da biblioteca pandas foi utilizado para importar os dados para um dataframe. Uma avaliação inicial indicou que não haviam dados nulos ou com tipos incorretos na planilha original, logo, não foi necessário fazer um tratamento para melhorar qualidade dos dados.

Antes de treinar o modelo, foram necessárias diversas etapas para prepará-los. A primeira foi uma manipulação do dataframe para armazenar os dados agregados e categorizados por ano, mês e região do Brasil.

Em seguida, houve uma manipulação do formato das datas para que elas ficassem igualmente distribuídas ao longo do período. Neste momento, também foi possível plotar os gráficos para cada região utilizando a biblioteca matplotlib e observar as características distintas de cada uma.

Na etapa de preparação para o treinamento, os dados foram separados em treino e teste com a função 'train_test_split' da biblioteca Scikit-Learn. Após alguns testes para determinar a melhor proporção entre os dois grupos, foi definido que dois terços dos dados seriam utilizados para treino e um terço para validação. Além disso, como a ordenação dos dados é importante para séries temporais, o parâmetro shuffle foi configurado como falso para não alterar a ordem do datafram original.

Ainda na etapa de preparação para o treinamento, foi utilizada a função 'fit_transform' da biblioteca MinMaxScaler do Scikit-Learn para normalizar os dados entre 0 e 1.

Como último passo na etapa de preparação, foi utilizada a função TimeseriesGenerator para determinar o formato de entrada dos dados no modelo de redes neurais e os valores que a rede deve tentar prever. Testes indicaram uma janela de entrada de 4 valores era a ideal como entrada para a rede neural.

Uma vez que os dados estavam prontos para o treinamento, restou apenas configurar as redes neurais para cada região. As redes foram configuradas em três camadas LSTM Bidirecional e uma de saída:

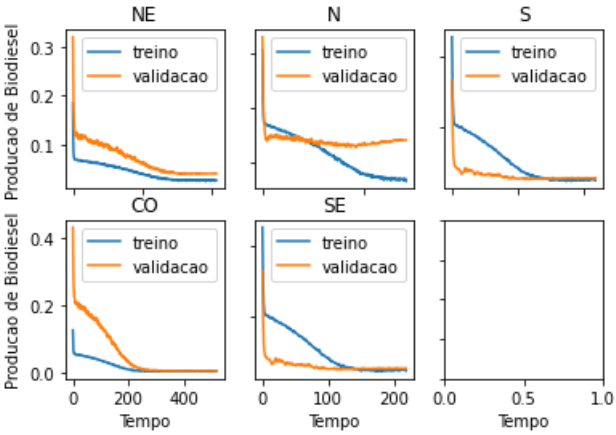
- LSTM Bidirecional com um dropout de 0.1 com tamanho 100.
- LSTM Bidirecional com um dropout de 0.1 com tamanho 80.
- LSTM Bidirecional com um dropout de 0.1 com tamanho 50.
- Saída (Dense) de tamanho 1.

Para compilar a rede, utilizou-se o otimizador SGD (Gradiente Otimizador Estocástico) e, além disso, o método do erro médio quadrático para o cálculo do para medir a proximidade da previsão com os dados de validação.

O treinamento foi configurado para 100 épocas, um batch size igual a 8, um percentual de validação de 20% e, por fim, um early stopping com paciência de 100.

3. Resultados

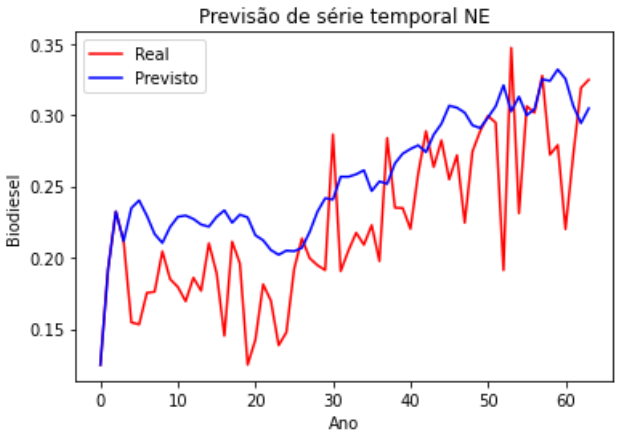
Os resultados variaram muito para cada região. A análise gráfica do erro ao longo do tempo indicou que algumas regiões chegaram rapidamente num ponto ótimo, enquanto outras tiveram uma dificuldade maior para convergir, como pode-se observar na figura abaixo:

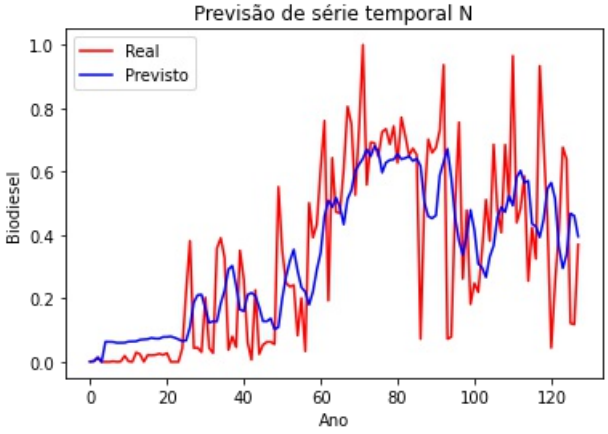


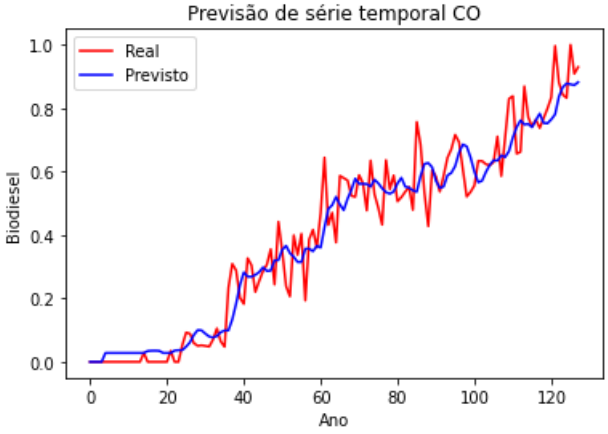
Já a análise do RMSE de todas as regiões ficou entre 0.05 e 0.2, valores considerados baixos.

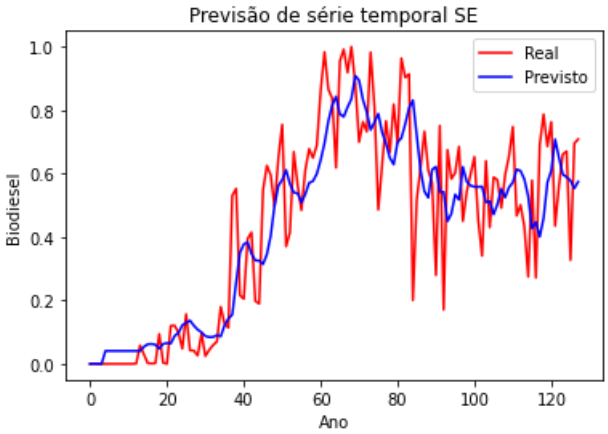
```
Train Score NE: 0.17 RMSE
Test Score NE: 0.05 RMSE
Train Score N: 0.20 RMSE
Test Score N: 0.20 RMSE
Train Score CO: 0.08 RMSE
Test Score CO: 0.08 RMSE
Train Score SE: 0.16 RMSE
Test Score SE: 0.16 RMSE
Train Score S: 0.08 RMSE
Test Score S: 0.08 RMSE
```

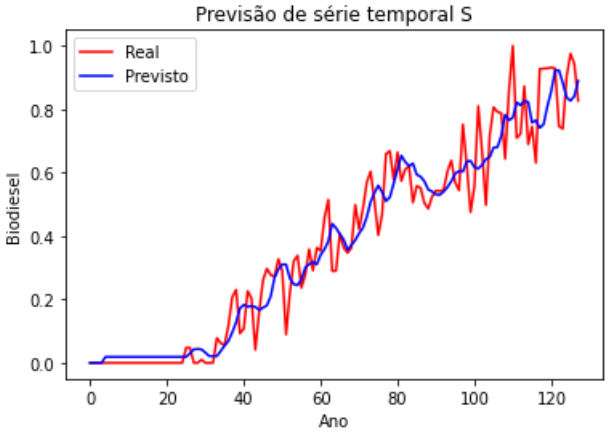
Contudo, apesar do erro pequeno e da capacidade de convergir rapidamente para um ótimo local, foi possível observar, de maneira gráfica, que os resultados poderiam ser melhores.











Como pode ser observado nos gráficos acima, o modelo foi capaz de prever bem a tendência das séries temporais. Entretanto, as oscilações previstas pelo modelo são muito maiores do que a dos dados fornecidos.

4. Conclusões

Embora o modelo seja capaz de detectar a tendência das curvas de cada região, apenas a utilização dos dados históricos de produção de biodiesel foi incapaz de prever variações esporádicas das curvas. Dessa forma, para futuros trabalhos, análise utilizando outras variáveis seria justificado para tentar melhorar o comportamento de previsibilidade da rede.

Matrícula: 191.671.006

Pontifícia Universidade Católica do Rio de Janeiro

Curso de Pós Graduação *Business Intelligence Master*