Aplicando Predição de Série Temporal Em Dados de Temperatura da Cidade de Dois Vizinhos

Gabriel Souza de Paula, Jober Guifor Campos, Willian Alberto Lauber

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Estrada para Boa Esperança, Km 04 CEP 85660-000 - Dois Vizinhos - PR - Brasil
COENS - Coordenação de Engenharia de Software

gabriel.paula@hotmail.com, jobergc@gmail.com, willianlauber@users.sourceforge.net

Resumo. *As séries temporais diferem dos problemas convencionais de aprendizagem de máquinas porque as observações dos dados não são independentes. Este artigo apresenta alguns tratamentos clássicos e uma maneira bastante simples de aplicar a aprendizagem em máquina em um contexto local, a previsão do tempo em Dois Vizinhos; cidade do estado do Paraná.*

1. Introdução

Conforme Overland Amaral a previsão do tempo é um dado essencial para o desenvolvimento econômico do país: se na agricultura pode determinar os rumos de uma plantação, na geração de energia em usinas hidrelétricas, que dependem do volume de água, pode ter impactos nas mais diversas atividades econômicas. Também contribuem para prevenir acidentes e prejuízos em setores de serviços e produção.[CreaSE]

O meteorologista explica que a previsão do tempo é uma ciência milenar que vem ganhando uma nova e inédita dimensão: a alta tecnologia que se usa hoje para entender as variáveis do clima e, especialmente, a necessidade de lidar com as mudanças climáticas.

Sendo o sudoeste do Paraná um grande produtor agrícola fica ressaltada a importância da previsão do tempo na economia do sudoeste desse estado[ALVES SANTOS], essa técnica também pode ter impactos na qualidade de vida como a diminuição do desgaste físico no trabalho agrícola devido a maior eficiência no plantio.

2. Objetivo Geral

Diante da importância para o mercado local da previsão do tempo este trabalho visa criar um modelo de aprendizagem de máquina utilizado redes neurais para a predição do tempo de uma cidade do sudoeste do Paraná, Dois Vizinhos.

2.1. Objetivos Específicos

Fazer uma simulação da predição do tempo para demostrar a eficiência e eficacia do modelo utilizando-se de uma métrica de qualidade.

3. Materiais e Métodos

Esta seção abordara as tecnologias e dados que foram utilizados na criação do modelo de predição e os aspectos relevantes em sua configuração e tem como intuito demostrar a credibilidade e reprodutibilidade do modelo.

3.1. Dados

Nosso banco de dados consiste em um conjunto de séries finitas que foram coletadas na Estação Meteorológica de Dois Vizinhos, pertencente ao INMET. Esses dados foram fomentados pelo GEBIOMET - Grupo de Estudos em Biometeorologia da UTFPR - Universidade Tecnológica Federal do Paraná -.

Para seu uso adequado os dados numéricos, salvo a data pois houve um tratamento na própria aplicação, foram transformados para o valor numérico correspondente na notação estadunidense, ou seja trocaram-se pontos por virgulas pois a biblioteca trabalha com valores nesse padrão americano.

3.2. Tecnologias

A linguagem python possui bibliotecas específicas para se trabalhar com aprendizagem de máquina. Neste trabalho empregaram-se algumas delas:

Datetime: É uma biblioteca que permite a manipulação de datas e sua transformação para dados ordinais.

Numpy: Permite a manipulação de matrizes, transformação dos dados, geração de números aleatórios e possui recursos que permitem se trabalhar com álgebra linear.

Matplotlib: Permite a visualização de dados em diversos gráficos.

Pandas: Extração e modelagem de dados de arquivos e análise exploratória de dados.

Scikit-learn: Permite a criação de modelos e sua execução para tarefas como predição, classificação e clusterização de dados, nessa biblioteca há também módulos de metrificação de modelos.

3.3. Metodologias

Para a elaboração do modelo foi empregada um algoritmo de aprendizagem de máquina supervisionado. Esse algoritmo realiza uma regressão em cima de dados contínuos, uma série temporal. Para a realização da predição de tempo em cada data foi aplicado uma janela deslizante em que, usando etapas de tempo anteriores como variáveis de entrada, vê-se no próximo passo de tempo uma variável de saída, ou seja: com base em tendencias anteriores é realizada uma predição do tempo para a próxima data, que nesse algoritmo corresponderá ao próximo dia.

3.4. Elaboração do modelo

Em aprendizagem de máquina as funções de ativação são responsáveis pela mutação nos pesos dos parâmetros dos neurônios da rede neural. O módulo *multilayer perceptron* para *regressor*, conforme a documentação da biblioteca scikit-learn, é uma rede neural que, apesar do nome, pode conter outros tipos de neurônios. Ela classifica ou prediz instâncias processando uma combinação linear de variáveis explanatórias.

Dentre as funções de ativação disponíveis (*'identity', 'logistic', 'tanh', 'relu'*) a função de ativação escolhida durante a elaboração do modelo foi a *tahn* pois

ela possui a melhor adequação aos dados. A funções e os MMRE's obtidos são apresentados a seguir:

tanh $\sigma z = \tanh z$: 0.178884073518;

logística $\sigma_z = \frac{1}{1+\exp(-z)}$; 0.178884153063;

rectified linear (ReLU): $\sigma_(z) = \max(0, z)$; 0.178884368848;

identity: $\sigma_z = z$; 0.178884375529.

Funções de ativação como a *identity* não são recomendados para este tipo de problema pois possuem uma alteração nos valores da série mais abrupta. Por exemplo a transformação *identity* transforma um valor intermediário em um extremo como 0 e 1 o que poderia deixar a predição com poucos dados próximo a valores máximos ou mínimos mesmo que os dados reais fossem intermediários e em predições com vários dados não realizar as curvas de forma suave podendo ocasionar um *underfiting* ou *overfiting*.

Utilizou-se uma taxa de aprendizagem adaptativa com vistas a acelerar a execução da predição com o modelo uma vez que havia uma quantidade razoável de dados.

Também foram empregadas 4 camadas neurais, duas escondidas, para adequar o modelo à grande variação dos dados pois mudam em uma taxa variável.

Existem funções de ativação mais adequadas para esse modelo como a *softmax* por terem uma curva de suavização menos aguda mas por não estarem na biblioteca, pelo menos para o módulo utilizado, não foram escolhidas.

O código resultante do modelo foi:

```
model = MLPRegressor(activation="tanh", solver='lbfgs',
                     hidden_layer_sizes=(10, 35, 53, 35),
                     max_iter=100, learning_rate = 'adaptive',
                     shuffle=True, random_state=1)
```

O cálculo dos peso de cada neurônio pode ser representado pela seguinte função:[Dpascual]

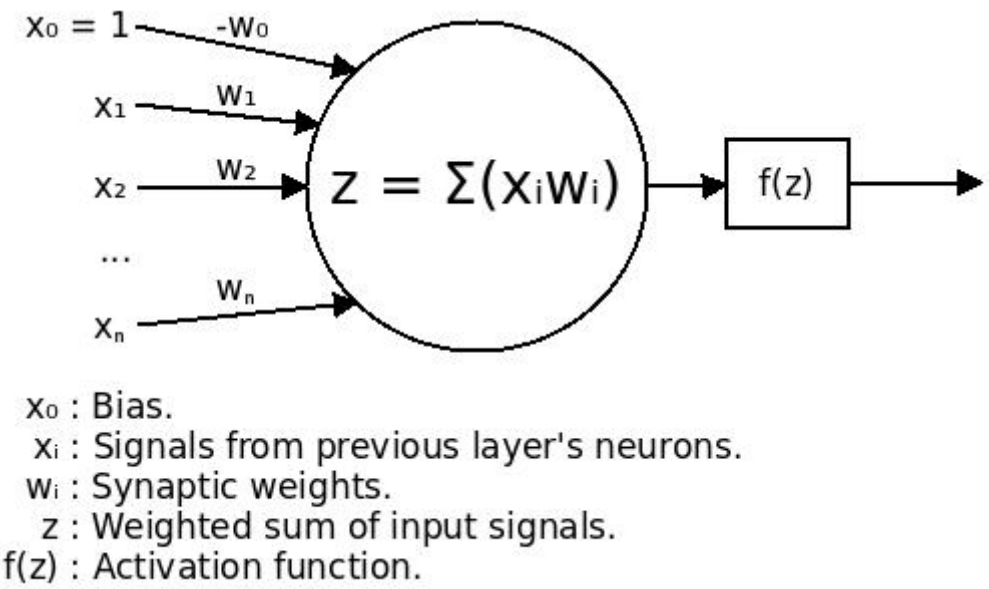

Figura 1. Fórmula utilizada no cálculo dos pesos para cada neurônio.

4. Simulação

A simulação contou com 13131 dados de entrada e foi realizada entre 5 e 6 de Dezembro de 2017.

4.1. Segmentação dos dados de aprendizagem

Para o treinamento da rede neural utilizamos todos os dados disponíveis, mas para a predição de cada item foram considerados apenas os valores que estavam dentro da janela deslizante isso pelo motivo de que na predição de valores contínuos valores muito antigos podem influenciar negativamente no resultado da predição atual.

Outrossim essas séries são muitas vezes infinitas e a previsão deve basear-se na análise estatística das evoluções anteriores por questões como custo computacional.

4.2. Validação do modelo

Para a validação dos modelos foram criados variáveis para armazenar os dados. As variáveis *X_train* e *y_train* são os dados de treinamento e utilizam todos os dados entre o item atual e os 10 itens posteriores. Já os dados de teste utilizam apenas os dados atuais.

A métrica de qualidade do modelo de predição foi feita em cima dos dados de teste, para cada dado predito foi calculado o MRE e um contador foi implementado para calcular o total de itens

```
end = dadosY.shape[0]
window = 10
totalItems, mre, x = 0, 0, 0

for i in range(1, end-window):

  print ("Iteração = " + str(i))
  X_train = dadosX[i:i+window]
  y_train = dadosY[i:i+window]

  x_test = dadosX[i+window]
  y_test = dadosY[i+window]

  model = MLPRegressor(activation="tanh", solver='lbfgs',
                       hidden_layer_sizes=(10, 35, 53, 35),
                       max_iter=100, learning_rate = 'adaptive',
                       shuffle=True, random_state=1)

  model.fit(X_train, y_train)
  x = model.predict([x_test])
  z = float(x) - float(y_test)
  if z < 0:
      z *= - 1
  mre += z/float(x)
  totalItems += 1
  y_pred.append(x)
  y_true.append(y_test)

# Transforma as listas em arrays numpy
# para facilitar os cálculos

y_pred = np.array(y_pred)
y_true = np.array(y_true)

print "MMRE(Mean Magnitude of Relative Error): " +
str(mre/totalItems)
```

5. Resultados

A métrica utilizada para a validação da qualidade foi o MMRE (Mean Magnitude of Relative Error). Seu calculo da-se da seguinte maneira:

Primeiro calcula-se o MRE (Magnitude of Relative Error) que é o módulo da diferença entre o valor esperado e o calculado em relação ao valor esperado. Na figura a seguir o divisor é o valor esperado e os y de cima o valor esperado e o valor calculado. A seguir é calculada a média entre todos os MRE's.

$$MRE = \frac{\left| y - y \right|}{y}$$

Figura 2. Fórmula utilizada no cálculo do MRE.

$$MMRE = \overline{MRE} = \frac{1}{n} \sum_{i=1}^{n} MREi$$

Figura 3. Fórmula utilizada no cálculo do MMRE.

O resultado obtido foi 0.178884073518. Com base na tabela a seguir pode-se classificar o modelo como excelente.

#	Classification	Range
1	Excellent	MMRE $\leq$ 0.368
2	Good	0.368 <MMRE $\leq$ 0.493
3	Fair	0.493 <MMRE $\leq$ 0.875
4	Negligible	MMRE >0.875

Figura 4. Critério para a classificação de modelos de predição. [SILVA]

Os gráficos resultantes do processamento do algoritmo foram:

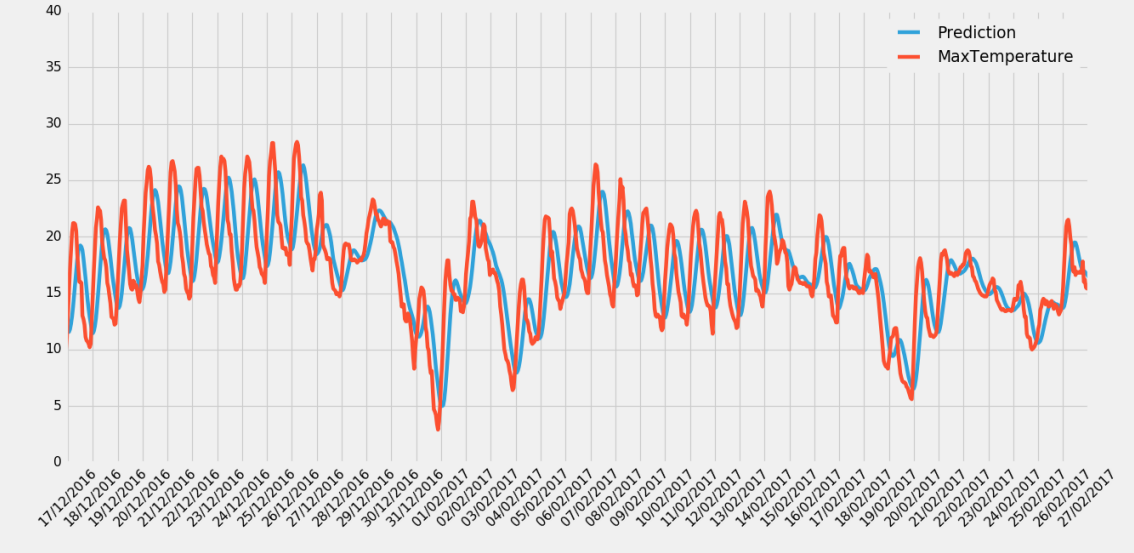

Figura 5. Predição da temperatura máxima dos últimos dias disponíveis na base de dados consultada. Fonte: os autores.

6. Conclusão e Perspectivas

Este trabalho apresentou a relevância da predição do tempo no contexto local e propôs um modelo que poderá ser utilizado em futuras predições do tempo com a mesma base de dados.

Descrevemos os passos para a normalização e segmentação dos dados utilizados nesse trabalho. Além disso foi apresentado um gráfico com a predição de uma série temporal.

Seguindo a métrica de MMRE pode-se classificar o modelo como excelente.

Para concluir, este trabalho abre várias perspectivas de novos trabalhos. Todas as tarefas que se utilizam de uma predição temporal, como plantio (por exemplo, para determinar a melhor época do ano para uma determinada espécie ser introduzida) ou criação de animais (por exemplo, "supressão de ruído da série temporal") pode ser abordada por nossa abordagem.

Além disso, a abordagem de tratamento de dados pode ser automatizada e poderia ser usada para um modelo para a predição do tempo em Dois Vizinhos em um sistema web e assim dar uma versão dos dados facilmente acessível aos leigos no assunto.

Finalmente, a introdução desse modelo abre o caminho para a extração de dados de estruturas temporais de outras saídas, como fluxos de texto ou resultados de cálculos.

Referências

ALVES SANTOS, R. Território e modernização da agricultura no Sudoeste do Paraná. . Disponível em: http://eduem.uem.br/ojs/index.php/EspacoAcademico/article/viewFile/11732/6709. Acesso em: 6/12/2017.

Crea-SE, A importância da meteorologia vai muito além de saber "se vai chover hoje". .Disponível em: <http://www.crea-se.org.br/a-importancia-da-meteorologia-vai-muito-alem-de-saber-se-vai-chover-hoje/>. Acesso em: 6/12/2017.

Dpascual-tfg, jderobot. .Disponível em: <http://jderobot.org/Dpascual-tfg>. Acesso em: 6/12/2017.

GEBIOMET - Grupo de Estudos em Biometeorologia. .Disponível em: <http://www.gebiomet.com.br/downloads.php>. Acesso em: 6/12/2017.

SILVA, G. C. Factors that Impact the Cloud Portability of Legacy Web Applications. , 2016. Disponível em: <http://etheses.whiterose.ac.uk/16428/8/Thesis - Corrections for deposit.pdf>. Acesso em: 7/12/2017.