Universidade Positivo
Disciplina: Sistemas Inteligentes
Profª Malga

Alunos: Caio Willian de Melo Runpfe
Felipe Gabriel Ferreira

Link do repositório: https://github.com/caiowmr/FlightDelayML.git
Link do DataSet: https://www.kaggle.com/datasets/divyansh22/february-flight-delay-prediction

DataSet Utilizado: Foi escolhido um conjunto de dados abrangente que inclui informações essenciais sobre voos, sendo as variáveis:

"DAY_OF_MONTH"

"DAY_OF_WEEK",

"OP_UNIQUE_CARRIER",

"OP_CARRIER_AIRLINE_ID",

"OP_CARRIER",

"TAIL_NUM",

"OP_CARRIER_FL_NUM",

"ORIGIN_AIRPORT_ID",

"ORIGIN_AIRPORT_SEQ_ID",

"ORIGIN","DEST_AIRPORT_ID",

"DEST_AIRPORT_SEQ_ID",

"DEST",

"DEP_TIME",

"DEP_DEL15",

"DEP_TIME_BLK",

"ARR_TIME",

"ARR_DEL15",

"CANCELLED",

"DIVERTED",

"DISTANCE".

Esses dados foram explorados para criar uma rede neural em Python, visando prever possíveis atrasos nos voos.

Variáveis escolhidas:

"DAY_OF_MONTH"

"DAY_OF_WEEK",

"OP_CARRIER_AIRLINE_ID",

"DEP_TIME",

"ARR_TIME",

"DEP_DEL15",

"DISTANCE".

Variável objetiva:

"ARR_DEL15",

Explicação do código:

O código implementa uma rede neural com **três camadas ocultas** para a tarefa de classificação, utilizando o algoritmo de retropropagação para treinamento. A rede foi projetada para prever atrasos de voos com os atributos: dia do mês, dia da semana, companhia aérea, horário de partida, horário de chegada, atraso de partida e distância do voo.

A função de ativação da tangente hiperbólica (tanh) é aplicada às camadas ocultas da rede para introduzir não linearidades nas transformações lineares dos dados. Isso permite que a rede capture relacionamentos mais complexos nos dados.

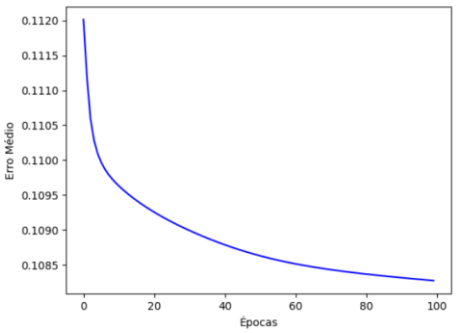
O treinamento da rede ocorre ao longo de várias épocas, ajustando os pesos das conexões entre as camadas para minimizar erros na previsão de atrasos na chegada de voos. Após o treinamento, o código realiza um teste de rede nos dados de entrada para avaliar sua capacidade de generalização.

Função de ativação: Tangente hiperbólica:

Teste 1:

Quantidade de épocas de treinamento: **1000**

Taxa de aprendizado: 0,01

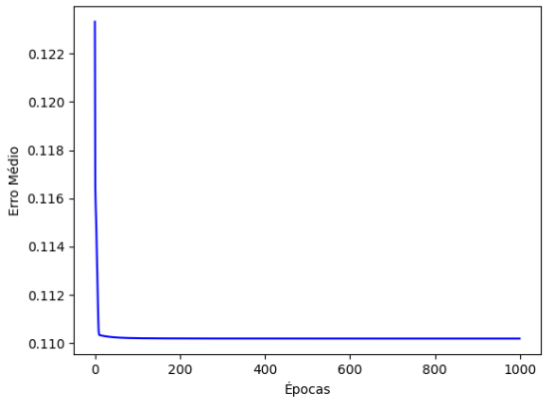



```
 [ 0.61468005]
 [ 0.61468005]
 [ 0.61468005]
 [-0.38531995]
 [ 0.61468005]
 [-0.38531995]
 [ 0.61468005]]
[[-0. -0. -1. ... -0. -1. -0.]
 [-0. -0. -1. ... -0. -1. -0.]
 [ 1.  1.  0. ...  1.  0.  1.]
 ...
 [ 1.  1.  0. ...  1.  0.  1.]
 [-0. -0. -1. ... -0. -1. -0.]
 [ 1.  1.  0. ...  1.  0.  1.]]
```

Teste 2:

Quantidade de épocas de treinamento: **1000**

Taxa de aprendizado: 0,07

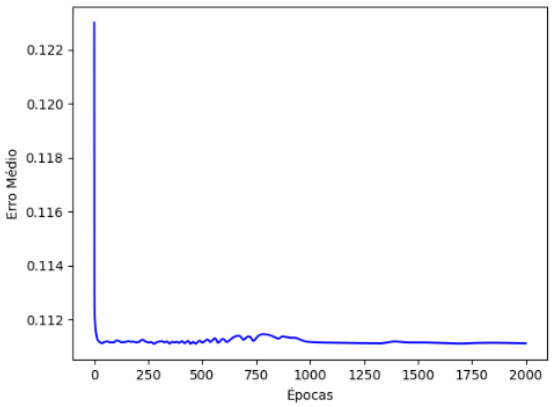



```
[ 0.59588237]
[ 0.59588237]
[ 0.59588237]
[-0.40411763]
[ 0.59588237]
[-0.40411763]
[ 0.59588237]]
[[-0. -0. -1. ... -0. -1. -0.]
[-0. -0. -1. ... -0. -1. -0.]
[ 1.  1.  0. ...  1.  0.  1.]
...
[ 1.  1.  0. ...  1.  0.  1.]
[-0. -0. -1. ... -0. -1. -0.]
[ 1.  1.  0. ...  1.  0.  1.]]
```

Teste 3:

Quantidade de épocas de treinamento: **2000**

Taxa de aprendizado: 0,09




```
[-0.42686246]
[ 0.57313449]
[ 0.57313449]
[ 0.57313754]
[-0.42686551]
[ 0.57313754]
[-0.42686551]
[ 0.57313754]]
[[-0. -0. -1. ... -0. -1. -0.]
 [-0. -0. -1. ... -0. -1. -0.]
 [ 1.  1.  0. ...  1.  0.  1.]
 ...
 [ 1.  1.  0. ...  1.  0.  1.]
 [-0. -0. -1. ... -0. -1. -0.]
 [ 1.  1.  0. ...  1.  0.  1.]]
```