Trainner - LAPISCO - 2020

Laboratório de Processamento de Imagens, Sinais e Computação Aplicada Instituto Federal do Ceará (IFCE)

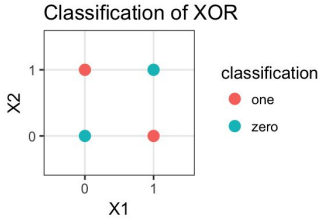
Professor: Aldísio Medeiros

Lista de Exercícios - Aula de Redes Neurais e Deep Learning

Instruções: Enviar em formato PDF as respostas para o email aldisio.medeiros@lapisco.ifce.edu.br com o assunto "TRAINNER2020 - LISTA - DEEPLEARNING".

Questão 1 - O que seria um problema não linearmente separável? Explique utilizando o problema da porta lógica XOR. Argumente sobre o motivo do Perceptron de Rosenblatt (1958) não conseguir resolver este tipo de problema.

Resposta 1: Um problema não linearmente separável é quando um plano como o bidimensional não pode separar as classes com apenas uma reta, como por exemplo a porta XOR, como representado no gráfico abaixo, não é possível separar as classes 1 e 0 apenas traçando uma reta.
O Perceptron de Rosenblatt não consegue resolver esse tipo de problema porque é uma rede neural para resolver problemas linearmente separáveis, ou seja, que pode separar as classes traçando uma reta, como exemplos as portas OR e AND. Porém, após alguns estudos ocorreu a combinação das portas lógicas anteriores, assim, cada lógica gerava uma reta, formando um cluster.



Questão 2 - Por que uma das estratégias para treinamento da rede neural é baseado no gradiente descendente e qual a relação do gradiente com a atualização dos pesos dos neurônios.

Resposta 2: A regra de aprendizagem do erro que serve para adaptar o peso da rede neural, se usa o gradiente porque o erro pode ser representado por uma função que vai crescer ou diminuir de acordo com erro, o método backpropagation é em cima do delta J (função de custo/erro) que depende diretamente do peso (w), que também depende da saída da rede.
A derivada parcial da saída que dependerá da variação da derivada do w. Quando tiver o resultado, é necessário minimizar, sendo assim, o vetor gradiente vai apontar para onde a função cresce, então como queremos minimizar, iremos apontar para o gradiente oposto, por isso é chamado de gradiente descendente.

Questão 3 - Considerando o dataset diabetes.csv, o mesmo utilizado nas atividades das aulas de machine learning, faça o treinamento e avalie as métricas de teste, para os modelos Shallow e Deep vistos em aula. Compare os resultados considerando as métricas: Acurácia, Tempo de treino e tempo de teste em relação aos classificadores MLP, KNN e QDA (vistos na aula de machine learning). Crie uma tabela com as métricas e liste os resultados, comente brevemente os resultados apresentados. Considere a Tabela 1 1 para listar os resultados.

Resposta 3: A acurácia de todos os modelos foram próximos, variando entre 65% e 78%, é possível dizer que o melhor modelo foi o QDA, porque a acurácia foi aproximadamente 78%, o tempo de treino e teste foram bem pequenas. Mas a acurácia do modelo shallow também foi 78%, porém como é um modelo mais simples, o tempo de treinamento é muito longo, sendo assim, se tivermos um dataset na escala de milhares, teremos um custo maior.

Classificador	Acurácia	Tempo de treino	Tempo de teste
Shallow Model	0.783550	8.061380	0.123559
Deep Model	0.653680	8.478806	0.128271
QDA	0.779221	0.001940	0.000460
MLP	0.766234	1.946205	0.000972
KNN	0.675325	0.002201	0.020878

Questão 4 - Considerando o dataset MNIST (importe a partir do keras.datasets), dataset dos dígitos manuscritos utilizado nos exemplos de CNN em aula. Faça o treinamento e avalie as métricas de treino e teste, para os modelos Fully Connected Model e Convolutional Neural Network, vistos em aula e compare os resultados considerando as métricas: Acurácia, Tempo de treino e tempo de teste em relação aos classificadores MLP, KNN e QDA (para estes classificadores, lembre-se de utilizar a vetorização da imagem). Crie uma tabela com as métricas e liste os resultados, comente brevemente os resultados apresentados. Considere a Tabela 1 para listar os resultados

Resposta 4: A acurácia foi boa na maioria dos métodos, exceto no QDA, porém o melhor método é o CNN que apresentou 98% de acurácia, mesmo com o tempo de treinamento muito alto, o nível de acerto é o melhor entre todos os métodos utilizados. Durante a compilação do código, o método KNN foi o que apresentou mais tempo para rodar o teste, levando cerca de 40 minutos para analisar o dataset.

Classificador	Acurácia	Tempo de treino	Tempo de teste
CNN	0.9810	292.341467	3.094650
Fully Connected Model	0.9778	49.763134	0.890145
QDA	0.1453	13.379785	1.433644
MLP	0.9386	26.891283	0.133963
KNN	0.9627	27.704790	1129.113036

Links úteis:

Download do dataset diabetes.csv

Jupyter com experimentos vistos em aula