Técnicas e Algoritmos em Ciência de Dados
Atividade avaliativa 2

Esta atividade deve ser submetida até 28 de junho de 2021 às 8h.
Submissões tardias NÃO serão aceitas.

Resultados de aprendizagem avaliados

Esta atividade testará os fundamentos de treinamento de redes neurais, agregação de árvores de decisão e implementação de florestas aleatórias.

Instruções

identificador

Escolha um número aleatório de 6 dígitos e escreva-o na primeira célula do notebook Python. Certifique-se de manter uma cópia desse número, pois ele será usado para fornecer o feedback (evite números triviais, como 000000 ou 123456, ou identificadores começando com zero – obrigado).

submissão

Envie seus arquivos através do ECLASS. Os arquivos que você envia não podem ser sobrescritos por mais ninguém, e eles não podem ser lidos por qualquer outro aluno. Você pode, no entanto, sobrescrever o arquivo enviado quantas vezes quiser, ressubmetendo-o, embora apenas a última versão enviada seja mantida. A submissão após o prazo NÃO será aceita.

Se você tiver problemas, no último minuto, envie sua atividade por e-mail como anexo para alberto.paccanaro@fgv.br com o tema *"URGENTE – ATIVIDADE 2 SUBMISSÃO". NO corpo da mensagem, explique o motivo para não enviar através do ECLASS.*

IMPORTANTE

- Sua submissão consistirá em um único notebook Python implementando suas soluções. Se você participar do desafio bônus (explicado abaixo), um arquivo zip contendo o notebook Python e o arquivo de predição deve ser enviado em seu lugar.
- O nome do arquivo será o número aleatório que identifica o aluno (por exemplo, 568423.ipynb)
- Esta atividade consiste em 4 partes. Certifique-se de que as 4 partes estão claramente separadas e identificáveis em seu Notebook Python.
- NÃO ENVIE NENHUM CONJUNTO DE DADOS, apenas o código.
- Qualquer função de utilidade que você usar deve ser incluída no notebook – não envie scripts utilitários.
- Notebooks Python podem ficar desordenados ao escrever o código. Se uma célula em seu notebook depender de outra célula que está ABAIXO dele, haverá desconto na nota.

CONSELHOS SOBRE EXERCÍCIOS BÔNUS

O valor de cada exercício em percentuais é fornecido -- a soma total dos pontos dos exercícios é de 100. Contudo, 10 pontos extras são dados para **exercícios bônus –** estes estão claramente indicados no texto abaixo**.** *Note que esses exercícios são difíceis e demorados. Aconselho você a ter uma solução final de todos os exercícios obrigatórios antes de tentar responder aos opcionais.*

Todo o trabalho que você submeter deve ser 100% de sua autoria. As submissões do curso serão verificadas contra plágio.

Critérios de pontuação

Esta atividade é avaliada e obrigatória e vale 25% da sua nota final total para este curso. Para obter pontuação máxima para cada pergunta, você deve respondê-la corretamente, mas também de forma completa. Pontos serão atribuídos para códigos bem estruturados.

TRABALHO DE CURSO

Parte 1 – Retropropagação – valor desta seção: 30 %

Baixe o arquivo "Part1.tsv" na plataforma ECLASS. Nesta parte, você construirá uma rede neural com uma arquitetura específica e a treinará com o algoritmo de retropropagação.

Aqui estão os passos que você precisará implementar:

a) Carregue os dados em um DataFrame pandas e obtenha um conjunto de dados compatível com scikit-learn.
b) Fazer uma divisão de 70%/30% do conjunto de dados para treinamento e teste, respectivamente.
c) Usando numpy, Implemente a rede neural da Figura 1. Observe que σ denota a função sigmóide, *tanh* denota a tangente hiperbólica e as setas verticais (simples) denotam o termo de *bias*. Você precisará implementar o algoritmo de retropropagação para aprender os pesos desta rede por gradiente descendente, ou seja:
    - O passe para a frente
    - O passe para trás
    - Atualizações dos pesos

    Nota: sua implementação deve contar apenas com numpy e não deve usar nenhuma estrutura especializada para retropropagação. Em outras palavras, seu código deve ser semelhante ao que vimos no laboratório de retropropagação
d) Avalie o modelo treinado usando:
    - precisão
    - AUC-PR
    - AUC-ROC
    - A matriz da confusão

    Nota: neste paso, você pode usar as ferramentas do scikit-learn, sem necessidade de implementá-las.
e) Durante cada época de treinamento, colete a perda, e faça uma figura com o número da época no eixo X, e a perda no eixo Y.

[Dica: devido à arquitetura específica que você deve implementar, isso não pode ser implementado de maneira direta usando o scikit-learn. Aconselho você a não perder tempo tentando.]

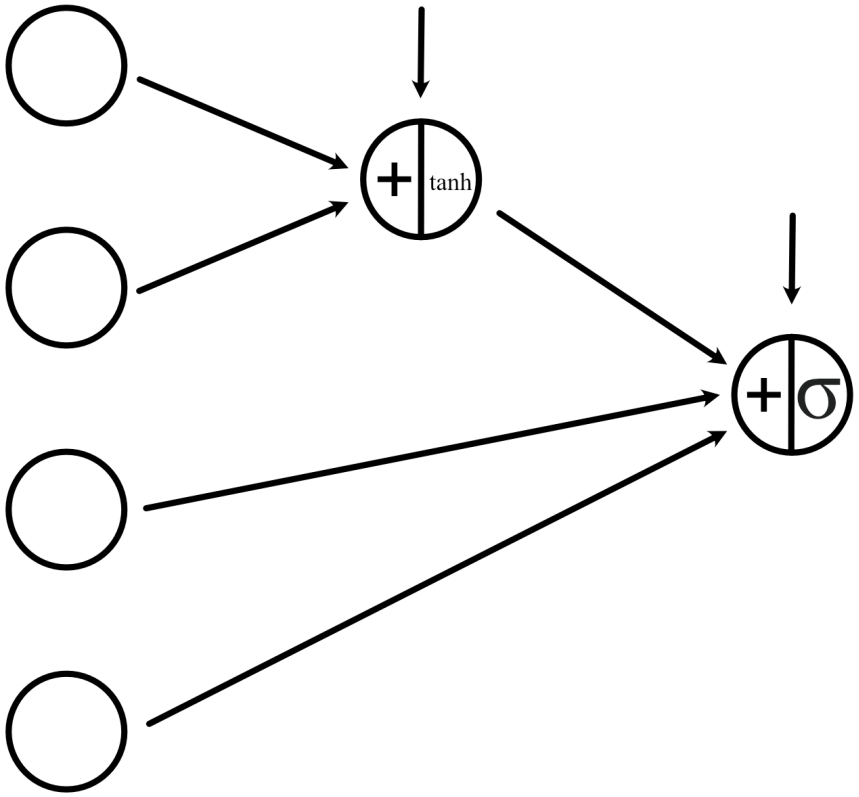

Figura1

Parte 2 – Agregação bootstrap – valor desta seção: 25 %

Baixe o Census Income dataset (https://archive.ics.uci.edu/ml/datasets/Adult). Nesta parte, você vai construir um modelo de árvores ensacadas para classificação.

Aqui estão os passos que você precisará implementar:

a) Carregue os dados em um DataFrame pandas e obtenha um conjunto de dados compatível com scikit-learn.
b) Fazer uma divisão de 70%/30% do conjunto de dados para treinamento e teste, respectivamente.
c) Implementar um modelo de árvores ensacadas. O modelo deve ter:
    - Um parâmetro B, o número de conjuntos de treinamentos bootstrap.
    - Parâmetros para controlar o crescimento das árvores, você precisa implementar pelo menos dois dos seguintes:
        i. Nível máximo da árvore
        ii. Número mínimo de observações em um nó
        iii. Proporção de classes no nó

[Dica: aqui, tome cuidado para que alguns dos recursos sejam numéricos (como o exemplo que vimos no laboratório) e alguns são nominais (ou categóricos).]

Parte 3 – Florestas Aleatórias – valor desta seção: 30 %

Esta parte usa os mesmos dados que você já usou na Parte 2.
Nesta parte, você implementará o algoritmo de Floresta Aleatória, como descrito na Seção 8.2.2 do livro Witten, James, Hastie & Tibshirani. Você deve adicionar as seguintes opções:

- Um parâmetro m para o número de preditores a considerar em cada divisão
- Um parâmetro para controlar o número de árvores na floresta
- Parâmetros para controlar o crescimento das árvores (poderia ser o mesmo da Parte 2)

Parte 4 – Comparação de desempenho – valor desta seção: 15 %

Esta parte conta com os modelos que você implementou nas Partes 2 e 3.

a) Use funções implementadas nas partes 2 e 3 para construir:
   - Um estimador scikit-learn de árvores ensacadas
   - Um estimador scikit-learn de florestas aleatórias
b) Usando o conjunto de dados que você construiu na Parte 2, treine ambos os modelos e compare os resultados usando as seguintes métricas de desempenho:
   - precisão
   - recall
   - AUC-ROC
   - AUC-PR

Seu código irá:

a) Imprimir uma tabela com valores da métrica de desempenho (colunas) para cada modelo (linhas) no conjunto de testes.
b) Fazer uma única figura contendo 2 subfiguras. Na primeira subfigura, você irá traçar as curvas ROC para cada modelo. Na segunda subfigura, você irá traçar as curvas de RP para cada modelo.

Parte 5 – BÔNUS DESAFIO – valor desta seção: 10 % DE NOTAS EXTRAS

O arquivo “Challenge-train.tsv” contém dados que descrevem associações entre proteínas e sua função. Para o propósito deste exercício, podemos pensar neste problema como um problema de classificação binária com 6 colunas, que você encontra no arquivo “Challenge-train.tsv”:

- As 5 primeiras são os features
- A última é o label para a classificação binaria (1 se a associação existe, ou 0 caso contrário)

O "Challenge-test.tsv" tem as primeiras 5 colunas descritas acima, mas o rótulo é omitido. Para este desafio, você usará qualquer método para treinar um classificador binário no arquivo “Challenge-train.tsv” e então o usará para gerar previsões para cada linha no arquivo

“Challenge-test.tsv”. Logo, você enviará suas previsões. Nós vamos testar suas previsões em relação aos valores verdadeiros e corretos.

- Se você submeter valores binários, 1 é considerado uma predição "positiva" (a associação existe), e 0 é considerado uma predição "negativa".
- Se você apresentar uma predição numérica, assumimos que a pontuação é maior quando o modelo prevê rótulos positivos, e menor quando o modelo prevê rótulos negativos.

Regras do desafio:

- Você envia um arquivo de texto simples chamado "prediction-<número>.txt", onde <número> é o número aleatório que você escolheu como seu identificador. Este arquivo deve conter um único número por linha.
- A primeira linha é a predição para a primeira linha em "Challenge-test.tsv"
- Se você enviar menos linhas do que as incluídas no arquivo de teste, assumiremos uma predição de 0 se o seu modelo prever o uso de etiquetas binárias, ou o valor mínimo, se o seu modelo prever usando uma pontuação numérica.
- Se você enviar mais linhas do que as incluídas no arquivo de teste, as linhas excedentes serão ignoradas.
- Você pode ser desqualificado se:
  - Você enviar mais de 1 arquivo de predição.
  - O formato do arquivo não é como descrito acima.
  - Se o formato do arquivo me impedir de abri-lo com um simples editor de texto.
  - O que você usou para nomear o arquivo não corresponde ao que você usa no notebook python enviado.

As notas serão alocadas da seguinte forma:

- As métricas AUC-ROC e AUC-PR serão calculadas
- Para a AUC-ROC e AUC-PR, respectivamente:
  - Os 20% melhores modelos terão notas extras de 5%
  - Os próximos 20% terão notas extras de 4%
  - . . .
  - Os últimos 20% terão notas extras de 1%

Dica 1: Quanto mais pessoas aderirem ao desafio, melhor para todos (pense nisso!)
Dica 2: É impossível encontrar esse conjunto de dados online, pois eu mesmo gerei esses valores.