Universidade Federal do Rio Grande do Sul **Inteligência Artificial**
Instituto de Informática **Prof. Anderson Rocha Tavares**
Departamento de Informática Aplicada

Trabalho 1 – Parte 3
Otimização por descida de gradiente

Instruções preliminares

As implementações devem ser feitas em Python 3. Caso precise de instalar bibliotecas adicionais (assuma que os códigos serão executados em uma maquina virtual com uma instalação padrão do Ubuntu e interpretador Python 3.8 com Miniconda, Pip, numpy e pandas), descreva as bibliotecas adicionais no seu Readme.md

1. Introdução

O objetivo do trabalho é explorar a implementação do algoritmo de gradiente descendente / descida de gradiente para regressão linear.

2. Tutorial

Abra o tutorial de regressão linear no link: https://colab.research.google.com/drive/1AEaWuxKgDOxOu_4dY7fHTAw2zwjo9mEG?usp=sharing

Faça uma cópia deste Colab para seguir o tutorial.

Para a parte de leitura, siga o passo a passo em cada célula, observando como os dados são lidos e organizados usando o numpy.

Sua implementação envolve completar o código das funções que `compute_cost`, que calcula o Erro Quadrático Médio e `step_gradient`, que realiza um passo da descida do gradiente, atualizando os parâmetros da regressão linear. Note que as células seguintes a cada uma dessas funções possuem código para testar sua implementação.

Observe que o Tutorial executa 10 épocas (iterações) de descida do gradiente, o que é mais que suficiente para resolver esse problema simples. Porém, ajuste esse número para sua implementação das atividades do item abaixo, usando o gráfico de Custo (Erro Quadrático Médio) por iteração para verificar a convergência da sua regressão.

3. Predição de preço de casas

Você irá implementar regressão linear para prever o preço de casas a partir de um conjunto de dados coletado em algum lugar dos Estados Unidos. O conjunto de dados, `house_prices_train.csv`, possui os dados de 1460 casas, uma por linha. O conjunto de dados possui 79 colunas, cada uma

medindo um atributo diferente das casas, mas vamos trabalhar apenas com algumas colunas. A última delas, `SalePrice`, é o preço de venda. Esse é o "y" que sua implementação deve prever, a partir dos atributos de uma casa "x".

Para começar, obtenha o conjunto de dados `house_prices_train.csv` , no Moodle.

3.1. Usando apenas a área da sala de estar

Nesta primeira parte, implemente a regressão linear para prever o preço de venda, `SalePrice`, usando apenas um atributo: a área da sala de estar (coluna `GrLivArea`).

Você deve normalizar o valor do atributo para todas as casas, usando o método min-max scaling, no qual o valor de um atributo é convertido para o intervalo [0, 1]. Percorra todas as linhas da tabela e atualize o valor usando a seguinte fórmula:

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}},$$

onde x é o valor do atributo na linha atual, x_{max} e x_{min} são os o valores máximo e mínimo do atributo em qualquer casa (ou seja, o maior/menor valor em toda a coluna), respectivamente, e x_{new} é o valor atualizado, que ficará no intervalo [0,1].

Não é necessário normalizar o preço de venda (`SalePrice`).

O motivo de começar com apenas um atributo é que você poderá usar as ferramentas de visualização do Tutorial para verificar os dados e também a sua implementação.

Implemente um script `launch_1attr.sh` que receba o conjunto de dados, o número de épocas de treinamento e escreva na tela os parâmetros: θ_0 (intercepto) e θ_1 (inclinação da reta), além do Erro Quadrático Médio obtido. Exemplo de execução do script para o `house_prices_train.csv` , com 100 épocas de treinamento (números fictícios, apenas observe o formato).

```
./launch_1attr.sh house_prices_train.csv 100
theta_0: 0.001
theta_1: 458.35999
Erro quadratico medio: 1268
```

3.2. Área da sala de estar e Qualidade Geral

Use sua implementação para prever o `SalePrice` usando a área da sala de estar (`GrLivArea`) e a Qualidade Geral (`OverallQual`).

Implemente um script `launch_2attr.sh` que receba o conjunto de dados, o número de épocas de treinamento e escreva na tela os parâmetros: θ_0 (intercepto) , θ_1 (parâmetro que multiplica `GrLivArea`), e θ_2 (parâmetro que multiplica `OverallQual`), além do Erro Quadrático Médio obtido. Exemplo de execução do script para o `house_prices_train.csv` , com 100 épocas de

treinamento (números fictícios, apenas observe o formato).

```
./launch_2attr.sh house_prices_train.csv 100
theta_0: 0.001
theta_1: 1E-5
theta_2: 23.441
Erro quadratico medio: 1068
```

3.3. Cinco atributos

Use sua implementação para prever o `SalePrice` usando a área da sala de estar (`GrLivArea`), a Qualidade Geral (`OverallQual`), a condição geral (`OverallCond`), a área da garagem (`GarageArea`) e o ano de construção (`YearBuilt`).

Implemente um script `launch_5attr.sh` que receba o conjunto de dados, o número de épocas de treinamento e escreva na tela os parâmetros: θ_0 (intercepto) ,θ_1, θ_2, θ_3 , θ_4, θ_5 e θ_6 (parâmetros que multiplicam `GrLivArea`, `OverallQual`, `OverallCond`, `GarageArea` e `YearBuilt`, respectivamente), além do Erro Quadrático Médio obtido. Exemplo de execução do script para o `house_prices_train.csv` , com 100 épocas de treinamento (números fictícios, apenas observe o formato).

```
./launch_5attr.sh house_prices_train.csv 100
theta_0: 0.001
theta_1: 0.23
theta_2: 489.667
theta_3: -852.001
theta_4: -59.36
theta_5: 1.98E-5
theta_6: -1.2E-7
Erro quadratico medio: 592
```

4. Entrega

A entrega deste trabalho será feita em um Questionário no Moodle, com algumas perguntas sobre seus resultados e um espaço para envio do código. Enviem apenas uma resposta por grupo (haverá uma entrada para colocar os dados dos componentes).

4.1. Respostas no Moodle

O primeiro item do Questionário é uma área de texto que deve ser preenchida com os nomes, cartões de matrícula e turma dos integrantes do grupo.

Além disso, para cada item da Seção 3, haverá questões para colocar os parâmetros encontrados pelo seu modelo e o Erro Quadrático Médio ao final do treinamento no `house_prices_train.csv`. Os valores serão comparados com uma implementação padrão.

4.2. Código

Na área para envio do código, dentro do Questionário de Entrega, você deve entregar um arquivo

`.zip` contendo:

- O código fonte completo.
- Os scripts `launch_1attr.sh, launch_2attr.sh` e `launch_5attr.sh`
- Um arquivo `Readme.md` em texto sem formatação com o seguinte conteúdo:
  - Nomes, cartões de matrícula e turma dos integrantes do grupo (mesmos preenchidos na área de texto do questionário);
  - Bibliotecas adicionais que precisem ser instaladas para executar sua implementação.

ATENÇÃO: os shell scripts (`.sh`) e o `Readme.md` devem se localizar na raiz do seu arquivo `.zip`! Isto é, o conteúdo do seu arquivo `.zip` deverá ser o seguinte:

```
launch1attr.sh
launch2attr.sh
launch5attr.sh
Readme.md
[arquivos do código fonte] <<pode criar subdiretórios se necessário
```

Conteúdo do arquivo `.zip` a ser enviado.

5. Observações gerais

- O trabalho deve ser feito em trios.
- Fiquem atentos à política de plágio!
- A nota final envolverá as respostas no moodle, mas tambem a execução do código: podemos chamar para esclarecimentos os trios cujas execuções derem resultados diferentes das respostas preenchidas no moodle, e ajustar as notas, caso seja necessário.

Política de Plágio

Trios poderão apenas discutir questões de alto nível relativas a resolução do problema em questão. Poderão discutir, por exemplo, questões sobre as estruturas de dados utilizadas, técnicas para visualizar os dados, etc. Não é permitido que os trios utilizem quaisquer códigos fonte provido por outros trios, ou encontrados na internet.

Pode-se fazer consultas na internet ou em livros apenas para estudar o modo de funcionamento das técnicas de IA, e para analisar o pseudo-código que as implementa. Não é permitida a análise ou cópia de implementações concretas (em quaisquer linguagens de programação) da técnica escolhida. O objetivo deste trabalho é justamente implementar as técnicas do zero e descobrir as dificuldades envolvidas na sua utilização para resolução do problema em questão. Toda e qualquer fonte consultada pelo trio (tanto para estudar os métodos a serem utilizadas, quanto para verificar a estruturação da técnica em termos de pseudo-código) precisa obrigatoriamente ser citada no Readme.md.

Usamos rotineiramente um sistema anti-plágio que compara o código-fonte desenvolvido pelos trios com soluções enviadas em edições passadas da disciplina, e também com implementações disponíveis online.

Qualquer nível de plágio (ou seja, utilização de implementações que não tenham sido 100% desenvolvidas pelo trio) poderá resultar em nota zero no trabalho. Caso a cópia tenha sido feita de outro trio da disciplina, todos os envolvidos (não apenas os que copiaram) serão penalizados. Esta política de avaliação não é aberta a debate. Se você tiver quaisquer dúvidas se uma determinada prática pode ou não, ser considerada plágio, não assuma nada: pergunte ao professor e aos monitores.

Note que, considerando-se os pesos das avaliações desta disciplina (especificados e descritos no Plano de Ensino), nota zero em qualquer um dos trabalhos de implementação abaixa muito a média final dos projetos práticos, e se ela for menor que 6, não é permitida prova de recuperação. Ou seja: caso seja detectado plágio, há o risco direto de reprovação. Os trios deverão desenvolver o trabalho sozinhos.