Miguel Fagundes Vuori

Um estudo de aprendizado ativo para tarefas de regressão

RELATÓRIO DE PROJETO FINAL

DEPARTAMENTO DE ENGENHARIA ELÉTRICA E DEPARTAMENTO DE INFORMÁTICA

Programa de graduação em Engenharia de Computação

Rio de Janeiro
Dezembro de 2022

Pontifícia Universidade Católica
do Rio de Janeiro

Miguel Fagundes Vuori

Um estudo de aprendizado ativo para tarefas de regressão

Relatório de Projeto Final

Relatório de Projeto Final, apresentado ao programa de Engenharia de Computação da PUC-Rio como requisito parcial para a obtenção do titulo de Engenheiro de Computação.

Orientador: Prof. Eduardo Sany Laber

Rio de Janeiro
Dezembro de 2022

Resumo

Vuori, Miguel Fagundes; Laber, Eduardo Sany. **Um estudo de aprendizado ativo para tarefas de regressão**. Rio de Janeiro, 2022. 30p. Projeto de Graduação – Departamento de Engenharia Elétrica e Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Aprendizado ativo, ou *active learning* (AL) em inglês é uma técnica dentro da área de aprendizado de máquina bastante útil quando se possui uma base de dados com grande número de amostras não rotuladas no qual o custo de rotulagem é alto, seja em questões financeiras ou de complexidade. Essa técnica se baseia em escolher exemplos para serem rotulados que melhorem a performance do modelo relativo à tarefa de aprendizado de máquina em questão. Esse trabalho visa estudar, implementar e analisar diferentes técnicas de AL para tarefas de regressão.

Palavras-chave

aprendizado de máquina, regressão, aprendizado ativo

Sumário

1
Introdução

1.1
Contexto

Em tarefas de aprendizado supervisionado é necessário uma grande quantidade de dados anotados para se obter uma boa performance no modelo à ser treinado. Esse processo de anotação de dados podem variar de complexidade dependendo da natureza dos dados, algumas rotulagens são feitas de forma muito rápida e sem custo, como por exemplo a utilização do sistema de *Recaptcha* criado por , no qual a anotação é feita de graça pelo usuário como forma de proteção contra bots em sistemas de login. Nele são apresentadas duas imagens de texto no qual, o usuário deve fazer a correspondência do que está escrito, sendo uma imagem usada para verificação e a outra usada pra reconhecimento da palavra escrita.

Porém para alguns projetos a rotulagem se torna muito custosa. No caso de prever emoções humanas alguns estudos(DEAP 2019) demonstram um trabalho muito maior para se dar rótulos, precisando de até 16 pessoas para se avaliar certa emoção.

De conhecimento dessa dificuldade, para tarefas que necessitam dessas variáveis independentes como o caso de tarefas de regressão, o aprendizado ativo se faz bastante útil. Aprendizado ativo é uma subárea dentro da área de aprendizado de máquina, responsável por ativamente selecionar dados que fornecerão uma melhor performance para o aprendizado de um modelo em questão.

1.2
Contribuições

Esse trabalho visa estudar, implementar e comparar diferentes técnicas de aprendizado ativo, especificamente na área de regressão linear, e assim analisar e validar resultados já encontrados na literatura.

1.3 Organização do trabalho

O trabalho será dividido em 4 partes. Na primeira parte, no capitulo 2, será fornecida uma apresentação de conceitos que serão importantes para desenvolvimento dos algoritmos de aprendizado ativo. Como aprendizado ativo para tarefas de regressão será o foco do trabalho, o primeiro conceito a ser apresentado será sobre regressão. O segundo conceito será sobre aprendizado ativo seguido pela integração dos dois conceitos, aprendizado ativo para regressão. Bootstrap e k-means clustering são utilizados em alguns algoritmos, então é importante apresentar esses também.

No capitulo 3 será apresentado os algoritmos estudados e implementados. Tanto um descrição em alto nível, complexidade como pseudocódigo serão fornecidos para um melhor entendimento do trabalho.

Os resultados e as análises do estudo serão apresentados no capitulo 4 bem como o ambiente no qual o estudo foi desenvolvido.

No último capitulo uma conclusão do estudo será fornecida.

2
Background

2.1
Regressão

Regressão é uma ferramenta dentro da área de estatística que é utilizada em diversas áreas do conhecimento, como por exemplo ecologia e financias. Seu uso possui como principais objetivos observar a correlação entre uma variável dependente e uma ou mais variáveis independentes, e obter a previsão da variável dependente através de novas instâncias das variáveis independentes(Beale Regression).

Porém existem diversos tipos de regressão. A mais comum e simples entre elas é a regressão linear, que se baseia em criar uma reta (no caso de 2 dimensões) ou um hiperplano (mais de duas dimensões) que melhor represente a distribuição dos dados no espaço, utilizando-se de em alguma expressão matemática para estimar os parâmetros do modelo, como por exemplo, o método dos mínimos quadrados comuns. O modelo de regressão linear pode ser descrito da seguinte forma matricial(gross2003linear):

$$y = X\beta + \epsilon,$$

$$y = \begin{pmatrix} y_1 \\ . \\ . \\ . \\ y_n \end{pmatrix}, X = \begin{pmatrix} X_{1,1} & . & . & . & X_{1,p} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ X_{n.1} & . & . & . & X_{n,p} \end{pmatrix}, \beta = \begin{pmatrix} \beta_1 \\ . \\ . \\ . \\ \beta_p \end{pmatrix}, \epsilon = \begin{pmatrix} \epsilon_1 \\ . \\ . \\ . \\ \epsilon_n \end{pmatrix}$$

Onde y é a matriz que representa a variável dependente e X a que representa as variáveis independentes. Cada linha representa uma amostra. O vetor β são os parâmetros do modelo, que, geralmente, se quer conhecer em tarefas de regressão. Já ϵ é a representação da variável não observável resultante da acumulação de erros. Essa variável é inclusa devido ao fato do modelo ser apenas uma aproximação linear e não uma representação linear exata(gross2003linear).

2.2 Aprendizado ativo

Técnicas de aprendizado ativo podem ser utilizadas para reduzir a quantidade de dados apresentadas no aprendizado de certo modelo. Elas se baseiam na hipótese de que o modelo teria uma performance melhor, com menos treino, se pudesse escolher os dados de treino. Algoritmos de aprendizado ativo foram desenvolvidos para se fazer a melhor escolha baseado em algum critério. Um fluxo bem comum de aprendizado ativo pode ser visto na figura 2.1 abaixo(settles2009active).

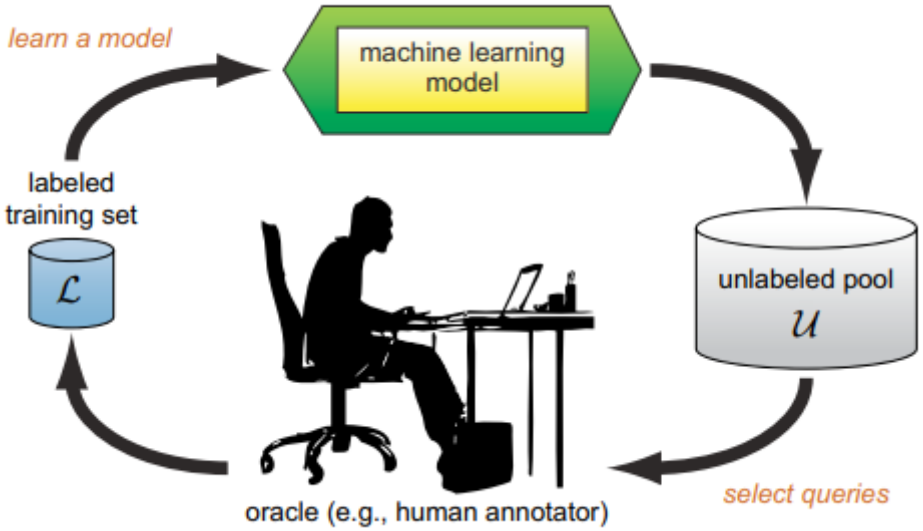

Figura 2.1: Ciclo de aprendizado ativo *pool-based*

Nesse esquema inicia-se com um conjunto de dados não rotulados $\mathcal{U}$ e um pequeno conjunto de dados já rotulado $\mathcal{L}$, seleciona-se um(modo sequencial) ou mais dados(*batch-mode*) para ser anotado pelo oráculo. Diversas técnicas podem ser utilizados na escolha desses dados. É então incrementado o conjunto de dados anotados e atualiza-se o modelo. Depois reinicia-se o ciclo até uma condição de parada, como por exemplo um limite de dados anotados ou então até o modelo atingir uma certa métrica desejada(settles2009active).

Existem diversos cenários no qual *active learning* pode ser utilizado. Um dos mais vistos, que será o caso de estudo desse trabalho, é o *pool-based active learning*. Esse problema é o mesmo caso do exemplo relatado no paragrafo acima, no qual se possui um conjunto de dados não rotulados(settles2009active).

2.3 Aprendizado ativo para regressão

A maior parte dos estudos de aprendizado ativo se concentram em tarefas de classificação, apenas uma quantidade reduzida é focada na parte de regressão(Dong 2019)(Wenb 2017). Diante desse cenário, esse estudo é focado em tarefas de regressão. Os primeiros algoritmos implementados serão os apresentados no artigo (Wenb 2017) e os últimos estão presentes no artigo (Dong 2019).

2.4 Bootstrap

Boostrap é uma técnica dentro da área de estatística com o objetivo de estimar uma medida, como por exemplo a média e desvio padrão, de uma população estatisticamente pequena. Dada uma população inicial com N amostras, a técnica se baseia em criar outras populações de tamanho N a partir da população inicial utilizando uma abordagem de amostragem com reposição. O passo a passo da técnica pode ser observado abaixo.

1. Escolhe-se o tamanho da população inicial e a quantidade de populações a serem geradas. Todas as outras populações geradas devem ter o mesmo tamanho.

2. Escolhe-se aleatoriamente uma amostra da população inicial com reposição.

3. Repita o processo anterior até que o tamanho da população seja alcançado. Ao fim desse item obtém-se uma população.

4. Repita o item 2 até que a quantidade de populações seja alcançado. Ao fim desse item obtém-se todas as população.

5. Calcula-se a métrica desejada de cada população e obtém-se a média da métrica.

Bootstrap também pode ser usado para se obter a saída de um modelo de predição através dos seguintes passos:

1. Escolhe-se o tamanho da população inicial(N) e a quantidade de populações a serem geradas(P). Todas as outras populações geradas devem ter o mesmo tamanho.

2. Escolhe-se aleatoriamente uma amostra da população inicial com reposição.

3. Repita o processo anterior até que o tamanho da população seja alcançado. Ao fim desse item obtém-se uma população.

4. Repita o item 2 até que a quantidade de populações seja alcançado. Ao fim desse item obtém-se todas as população.

5. Usando as P populações treina-se P modelos de predição.

6. Calcula-se a média da saída de cada modelo.

2.5 K-means clustering

K-means é uma técnica de agrupamento de dados baseado em suas similaridades. É um método bastante popular dentro da área de aprendizado não supervisionado. Dado uma base de dados de tamanho N, inicialmente, o algoritmo divide o espaço amostral em k clusters, e assume-se que cada amostra pertença ao cluster cujo centroide esteja mais próximo dessa amostra. Depois calcula-se o novo centroide de cada cluster e o processo se repete até que convirja.

Alguns dos benefícios da utilização do K-means é sua facilidade de entendimento e implementação. Além disso ele pode ser utilizado com uma variedade imensa de datasets. Algumas vantagens e desvantagens estão listadas abaixo.

Vantagens:

- Rapidez e desempenho em tempo de processamento.
- Fácil entendimento e utilização.
- Pode ser utilizado na maioria dos datasets, gerando resultados confiáveis.

Desvantagens:

- Ele assume que os dados estão agrupados em esfera.
- Sensível à uma boa escolha de k.
- Não é adequado para dados com outlier ou ruído.

Abaixo, na figura 2.2 é fornecido um exemplo de algumas iterações do algoritmo. Observe que o algoritmo é inicializado com $k = 2$ e com seus centroides escolhidos aleatoriamente. Em c calcula-se a distância euclidiana dos centroides à cada ponto e atribui cada ponto ao cluster mais próximo. Em seguida calcula-se o novo centroide para se repetir o passo anterior. Isso se repete até que aja a convergência em f.

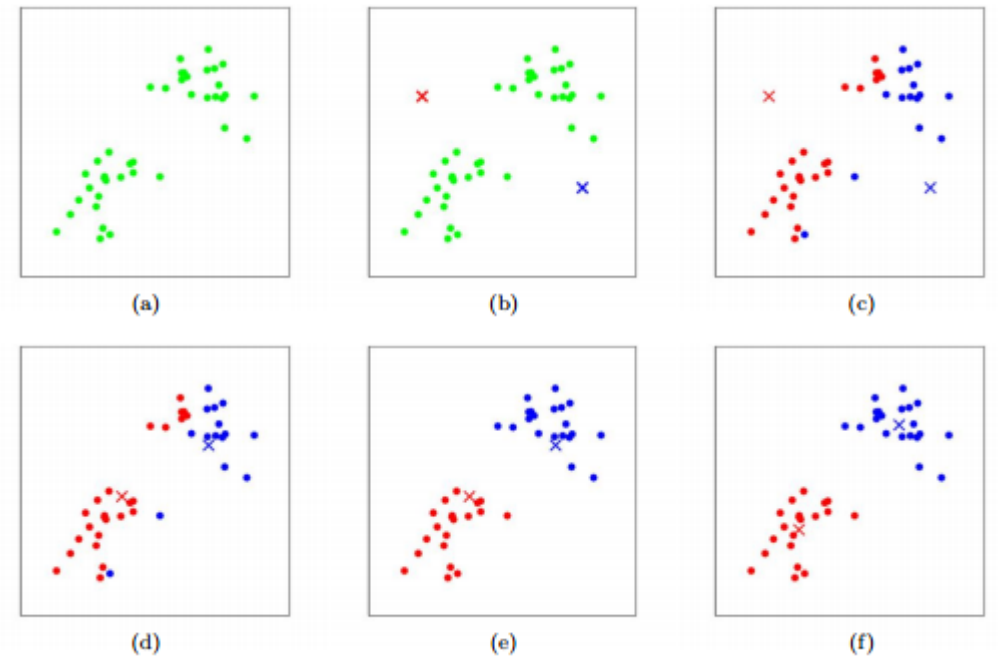

Figura 2.2: Iterações de K-means clustering

3
Algoritmos implementados

Neste capítulo apresentamos os algoritmos que foram implementados, começando pelos mais comuns: *Greedy Sampling*(GS) e *Expected Model Change Maximization*(EMCM). Em seguida será apresentado uma possível melhoria nos algoritmos anteriores utilizando uma abordagem sugerida no *paper* (Dong 2019). Por fim, será implementada uma alternativa para o algoritmo *Expected Model Change Maximization* em *batch-mode*.

3.1
Descrição dos Algoritmos

3.1.1
Expected Model Change Maximization(EMCM)

Esse algoritmo é utilizado tanto para tarefas de classificação como para tarefas de regressão. A ideia base do algoritmo é escolher o dado que mais irá mudar o modelo caso ele seja inserido na *pool* de dados rotulados, pois ele será considerado aquele que possui mais informação. É possível qualificar a mudança do modelo segundo o gradiente da função de perda relativo a cada instância de amostra candidata. A equação para se escolher o próximo dado candidato está formulada em (Wenb 2017) e sua dedução está apresentada abaixo.

Uma vez que o objetivo da regressão linear é gerar uma reta que melhor represente os dados podemos representar a saída do modelo como:

$$f(x;\theta) = \sum_{i=0}^{P} \theta_i x_{(i)} = \theta^T x \tag{3-1}$$

onde θ é o vetor de parâmetros do modelo de regressão, $x_{(i)}$ são os atributos da amostra x e $x_{(0)} = 1$ termo de interceptação. Seja o conjunto de dados de treino $D = (x_i, y_i)_{i=1}^{n}$, o modelo de regressão é ajustado para minimizar a função de custo do erro quadrático:

$$\hat{\epsilon}_D = \frac{1}{2} \sum_{i=1}^{n} (f(x_i) - y_i)^2 \tag{3-2}$$

O erro empírico acumulado com a adição de uma nova amostra ao conjunto de dados de treino se torna:

$$\hat{\epsilon}_{D^+} = \frac{1}{2}\sum_{i=1}^{n}(f(x_i) - y_i)^2 + \frac{1}{2}(f(x^+) - y^+)^2 \tag{3-3}$$

onde x^+ é uma amostra candidata com *label* y^+ e $D^+ = D \cup (x^+, y^+)$.

Como o objetivo do algoritmo de EMCM envolve escolher a amostra que produzirá maior mudança no modelo, calcula-se a derivada da função de erro $\mathcal{L}_{x^+}(\theta)$ em relação aos parâmetros θ em x^+.

$$\begin{aligned}\frac{\partial \mathcal{L}_{x^+}(\theta)}{\partial \theta} &= (f(x^+) - y^+)\frac{\partial f(x^+)}{\partial \theta} \\ &= (f(x^+) - y^+)\frac{\partial \theta^T x^+}{\partial \theta} \\ &= (f(x^+) - y^+)x^+\end{aligned} \tag{3-4}$$

Como y^+ é desconhecido à priori, a técnica de bootstrap é utilizada para gerar outros modelos de predição $\{f_1, f_1, \ldots, f_Z\}$, onde Z é o tamanho do conjunto, e assim se estimar a distribuição de $y^+ \in \{y_1, y_1, \ldots, y_Z\}$. Ao final escolhe-se a amostra relativa ao maior gradiente para ser rotulada:

$$x^* = \arg\max_{x \in U} \frac{1}{Z}\sum_{z=1}^{Z} \|(f(x) - y_z(x))x\| \tag{3-5}$$

no qual $y_z(x), z = 1, \ldots, Z$ são as saídas dos modelos treinados utilizando bootstrap e $f(x)$ é saída do modelo de predição corrente.

Dessa forma, EMCM escolhe o dado que possui mais informação(EMCM). Um pseudocódigo pode ser observado abaixo.

Algorithm 1 Expected Model Change Maximization(EMCM)

1: Seja N o dataset, $N - M_0$ o conjunto dos dados não rotulados, x^* o dado retornado pelo algoritmo e P o tamanho do conjunto de modelos a ser gerado.
2: Constrói-se um modelo linear utilizado os dados rotulados iniciais.
3: Constrói-se P modelos lineares utilizado *bootstrap*
4: $x = None, g = -\infty$
5: **for** $n = |M_0| + 1, \ldots, |N|$ **do**
6: Computa-se x^* usando a equação (3-5)
7: **end for**

3.1.2 Greedy Sampling(GS)

Greedy sampling é um algoritmo de *passivo*, ou seja, a escolha do dado é independente do modelo logo não há a necessidade de atualização do modelo a cada iteração. Ao contrário do último algoritmo mencionado que prioriza a informatividade na escolha do dado, GS faz a escolha do dado levando em conta sua geometria no espaço de características, isso prioriza a diversidade na escolha. Mais especificamente para cada dado não rotulado

x_n de $N - M_0$ é computado sua distância euclidiana até os dados de M_0 já rotulados(Dong 2019).

$$d_{nm} = |x_n - x_m|, n = 1, ..., |M_0|, m = |M_0| + 1, ..., |N| \tag{3-6}$$

Depois computa-se a menor distância $\underline{d}_n$ relativo à cada não rotulado.

$$\underline{d}_n = \min_m d_{nm}, n = 1, ..., |M_0| \tag{3-7}$$

Ao final escolhe-se o dado com maior $\underline{d}_n$. Abaixo está um possível pseudocódigo.

Algorithm 2 Greedy Sampling(GS)

1: Seja N o dataset, $N - M_0$ o conjunto dos dados não rotulados, e x o dado retornado pelo algoritmo.
2: Calcula-se as distâncias entre cada amostra.
3: $x = None, \underline{d}_n = -\infty$
4: **for** $n = |M_0| + 1, \ldots, |N|$ **do**
5: **for** $m = 1, \ldots, |M_0|$ **do**
6: Escolhe-se a menor distância d entre a amostra n e m
7: **end for**
8: **if** $d > \underline{d}_n$ **then**
9: $\underline{d}_n \leftarrow d$
10: $x \leftarrow n$
11: **end if**
12: **end for**

3.1.3 RD ALR algorithm

Em (Dong 2019) foi proposto uma abordagem que complementa os dois últimos algoritmos adicionando a preocupação da representatividade e diversidade na *query* do dado. Assume-se que a base de dados de cardinalidade $|N|$ está inicialmente não rotulada. Sua inicialização consiste em aplicar k-means cluster com $k = d$, sendo d a dimensão dos dados, e escolher os dados mais pertos do centroide para primeira rotulagem, isso garante a construção de um modelo linear minimamente razoável. Além disso, isso garante a representatividade pois cada dado escolhido é um bom representante do cluster a ser escolhido, e também garante a diversidade pois d clusters cobre boa parte do espaço amostral. Após a inicialização do algoritmo, na etapa iterativa no qual é escolhido um dado para rotulagem a cada iteração, é realizado novamente k-means clustering com $k = d + 1$. É possível notar que pelo menos 1 cluster não possuirá dados rotulados, uma vez que só existem d dados

rotulados. Então escolhe-se o maior cluster que não possui dados rotulados como sendo o mais representativo dentre os demais. No algoritmo básico de RD ALR a amostra mais perto do centroide é escolhida para ser rotulada, porém é possível integrar com as duas últimas abordagens já mostradas, ou seja, GS e EMCM para se fazer a *query* de uma amostra pertencente à esse cluster. Após *query* desse dado, repete-se o processo iterativo com $k = d + 2$ até um certo número de iterações ser alcançado. Um pseudocódigo com as 3 variantes implementadas é fornecido abaixo.

Algorithm 3 RD ALR algorithm e suas variações

1: Seja N o dataset, d a dimensão do dataset e M o número máximo de amostras para seleção.
2: Realiza-se k-means clustering com k = d e seleciona-se as amostras mais perto do centroide para a rotulagem inicial.
3: **for** $m = d + 1, \ldots, M$ **do**
4: Realiza-se k-means clustering com k = m.
5: Seleciona-se o maior cluster que não possui dados rotulados.
6: Opção 1: Seleciona-se a amostra mais perto desse centroide para rotulagem.
7: Opção 2: Seleciona-se a amostra utilizando GS para rotulagem.
8: Opção 3: Seleciona-se a amostra utilizando EMCM para rotulagem.
9: Adiciona-se essa amostra ao conjunto de dados rotulados
10: **end for**
11: Retorna-se um modelo de regressão com os dados selecionados e rotulados.

3.1.4 Batch-mode EMCM

Os algoritmos apresentados até então são considerados de *query* sequencial, uma vez que apenas uma amostra é escolhida para ser rotulada por iteração e o modelo é então atualizado a cada amostra. Nessa sessão será apresentada uma alternativa em batch para o algorítimo EMCM já apresentado, ou seja, um batch de amostras será escolhida a cada iteração e o modelo só será atualizado após o batch ser escolhido. Um diferencial a ser notado é que a escolha da j-ésima amostra do batch será influenciada pelas escolhas anteriores dentro do batch. *Batch Mode Active Learning for Regression With Expected Model Change*(EMCM) formula a seguinte equação para a escolha de cada batch a cada iteração:

$$x_j^* = \arg \max_{x \in U \setminus b_{j-1}^*} \sum_{z=1}^{Z} \frac{1}{Z} \|(f(x) - y_z(x))x - C(x, b_{j-1}^*)\|, j = 1, \ldots, k, 1 \leq i \leq j-1 \tag{3-8}$$

Onde x_j^* é o j-ésima escolha de dado do batch, j é o tamanho do batch, U o conjunto de dados não rotulados, b é o próprio batch, $f(x)$ é modelo de regressão linear, enquanto y é o conjunto de modelos de regressão linear treinados com a técnica de bootstrap. O termo $C(x, b_{j-1}^*)$ expressa a relação entre o candidato à escolha x e as amostras já escolhidas no batch b_{j-1}^*. Esse termo foi formulado em *Batch Mode Active Learning for Regression With Expected Model Change*(EMCM). Abaixo está o pseudocódigo relativo a esse algoritmo.

Algorithm 4 Batch-mode EMCM

Seja N o tamanho do dataset, $N - M_0$ o tamanho do conjunto dos dados não rotulados, b o batch de dados retornado e P o tamanho do conjunto de modelos a ser gerado.
Constrói-se um modelo linear utilizado os dados rotulados iniciais
Constrói-se P modelos lineares utilizado *bootstrap*
$b = \emptyset$
while $|b| < k$ **do**
    for $n = M_0 + 1, \ldots, N$ **do**
        Estima-se a mudança do modelo usando a equação (3-4)
    end for
    Seleciona-se n que gerará a maior mudança no modelo.
    $b \leftarrow b \cup n$
end while

4
Estudo Experimental

Um estudo experimental primário foi realizado utilizando alguns dataset já utilizados nos artigos mencionados. Esse estudo teve como intuito validar o funcionamento dos algoritmos implementados e comparando com os resultados já obtidos. A segunda parte do estudo visa aplicar as mesmas técnicas à bases de dados que não foram utilizadas antes.

4.1
Ambiente Computacional

O google coolaboratory foi a principal ferramenta utilizada para a implementação dos algoritmos. Ele é um serviço web fornecido pela google research, que permite a escrita e execução de código python. Ela foi baseada em notebooks hospedados do Jupyter, porém não é necessária nenhuma configuração inicial. Além disso, para execução dos notebooks, uma máquina virtual fornecida pela Google é utilizada. Nesse caso as especificações da CPU e GPU utilizadas estão apresentadas abaixo.

```
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 85
model name      : Intel(R) Xeon(R) CPU @ 2.00GHz
stepping        : 3
microcode       : 0x1
cpu MHz         : 2000.150
cache size      : 39424 KB
physical id     : 0
siblings        : 2
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
```

Figura 4.1: Especificações da CPU

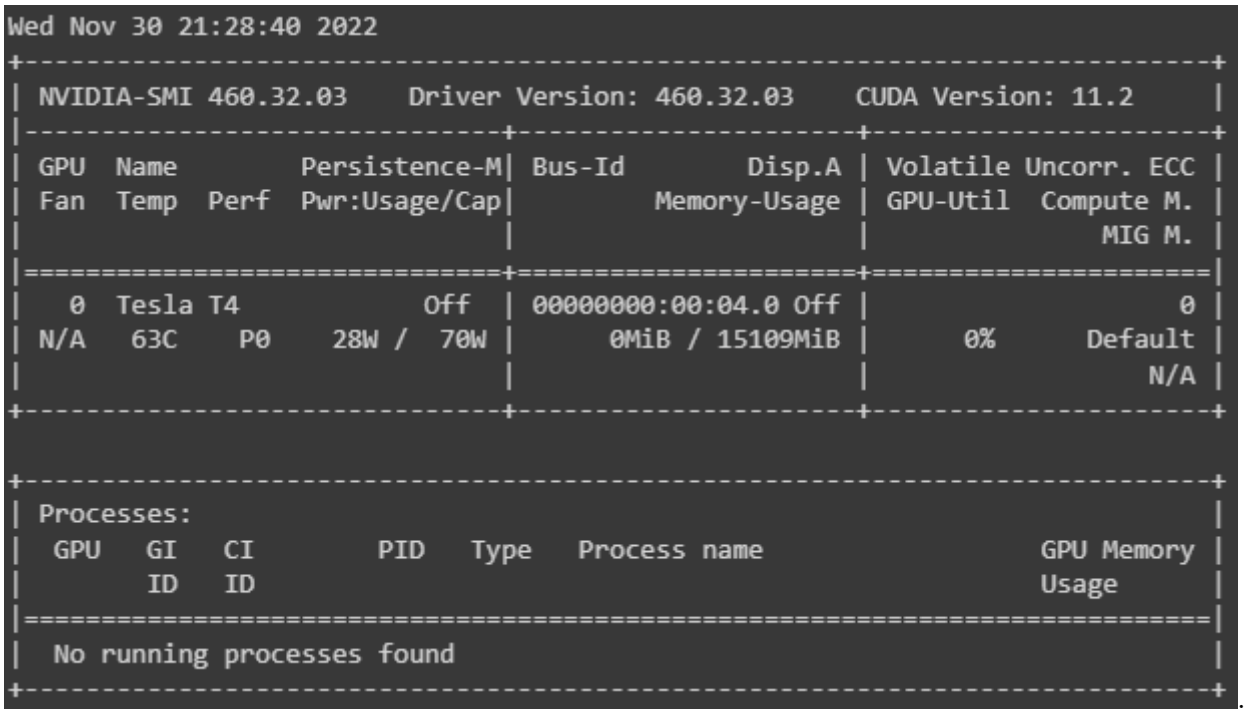

Figura 4.2: Especificações da GPU

4.2 Métricas

A cada iteração foram calculadas a raiz quadrada do erro quadrático médio(RMSE) e o coeficiente de correlação(CC) para se medir a performance de cada algoritmo. Uma vez que cada algoritmo possa fazer a query de diferentes amostras em uma mesma iteração se torna inviável comparar a performance baseada nas amostras não rotuladas. Dessa forma, as métricas escolhidas serão calculadas sobre toda a base de dados, sendo o rotulo das m amostras escolhidas seu verdadeiro rotulo, e os rótulos das $N - m$ amostras restantes a predição do modelo de regressão treinado. Dentre essas duas medições, RMSE é a principal métrica de avaliação, uma vez que a mesma é diretamente otimizada pelo modelo de regressão. Com o aumento do número de iterações é esperado que RMSE diminua, enquanto CC aumenta, porém não necessariamente. As equações de RMSE e CC estão apresentadas abaixo.

$$RMSE = [\frac{1}{N}\sum_{n=1}^{N}(y_n - y_n^{'})^2]^{1/2} \tag{4-1}$$

$$CC = \frac{\sum_{n=1}^{N}(y_n - \bar{y})(y_n^{'} - \bar{y}^{'})}{[\sum_{n=1}^{N}(y_n - \bar{y})^2]^{1/2}[\sum_{n=1}^{N}(y_n^{'} - \bar{y}^{'})^2]^{1/2}} \tag{4-2}$$

$$y_n^{'} = \begin{cases} y_n & \text{if } n = 1, \ldots, m \\ \hat{y}_n & \text{if } n = m+1, \ldots, N \end{cases} \tag{4-3}$$

$$\bar{y} = \frac{1}{N}\sum_{n=1}^{N} y_n; \bar{y}^{'} = \frac{1}{N}\sum_{n=1}^{N} y_n^{'} \tag{4-4}$$

Sendo y_n o verdadeiro rótulo de x_n e $\hat{y}_n$ a predição do modelo de regressão.

4.3 Base de dados

Os experimentos foram realizados em 13 diferentes base de dados obtidas em *UCI Machine Learning Repository* e em *Kaggle*. As tabelas autoMPG, concrete, NO2, Yacht e Housing foram utilizadas em estudos anteriores(Dong 2019)(Wenb 2017) e novamente foram testadas a fim de confirmar os resultados obtidos com esses dados.

As demais tabelas possuem como objetivo verificar se os resultados se mantém em outros datasets.

Como o projeto é focado em *active learning* para tarefas de regressão, todos os datasets possuem sua variável dependente contínua. Além disso foram escolhidas datasets multivariados e com categorias numéricas. Foi feita uma limpeza das tabelas para a retirada de amostras que possuíam categorias faltantes, e ainda um pré processamento também foi realizado: foi utilizado *One-Hot Encoding* nas colunas que possuem dados categóricos, a fim de transforma-las em numéricas, e todas as colunas foram normalizadas de tal forma que a média seja zero e o desvio padrão seja 1. Abaixo é fornecido o número de variáveis independentes e a quantidade de linhas de cada tabela.

Base de dados		
Tabela	Número de colunas	Número de linhas
autoMPG	7	392
concrete	8	1030
N02	7	500
Yacht	6	308
Housing	13	506
adm_data	6	400
Bodyfat	14	252
Fish	13	159
qsar_fish_toxicity	6	908
qsar_aquatic_toxicity	8	546
Machine	38	209
heart_failure	12	299
synchronous_machine	4	577

Tabela 4.1: Base de dados.

4.4 Baseline

Como baseline(BS) foi escolhida uma abordagem randômica no qual a cada iteração escolhe-se aleatoriamente uma amostra para se obter o *label*.

4.5 Resultados

Os resultados relativos à todos os algoritmos e datasets estão apresentados abaixo. Foram realizadas um total de 100 iterações em cada dataset. O algoritmo em *batch-mode* foi executado com tamanho de batch igual à 5. As curvas vermelhas, azuis, verdes, cianas, lilas, amarelas e pretas são relativas, respectivamente, aos algoritmos GS, EMCM, RDALR opção 1(seleciona-se a amostra mais perto do centróide), RDALR opção 2(integração com o algoritmo GS), RDALR opção 3(integração com o algoritmo EMCM).

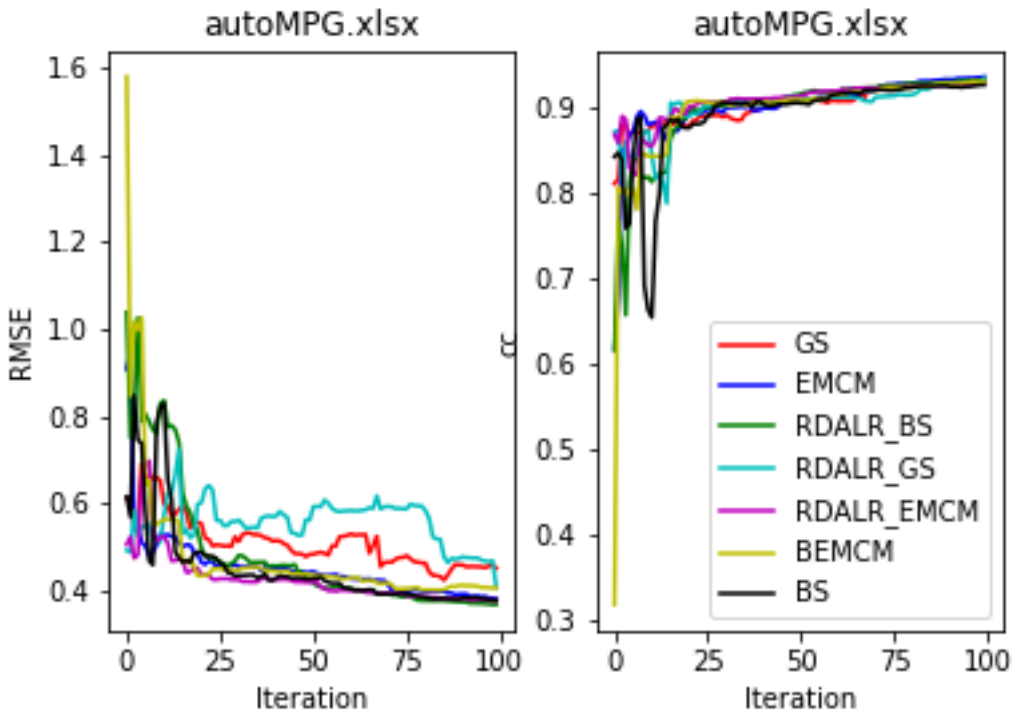

Figura 4.3: RMSE e CC de autoMPG

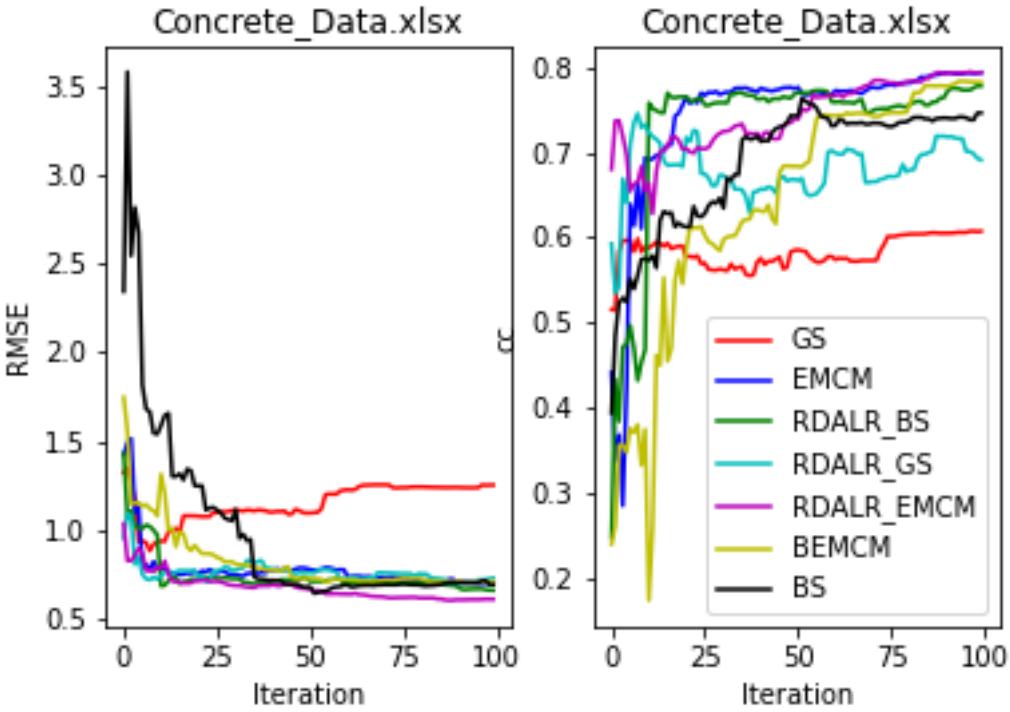

Figura 4.4: RMSE e CC de Concrete

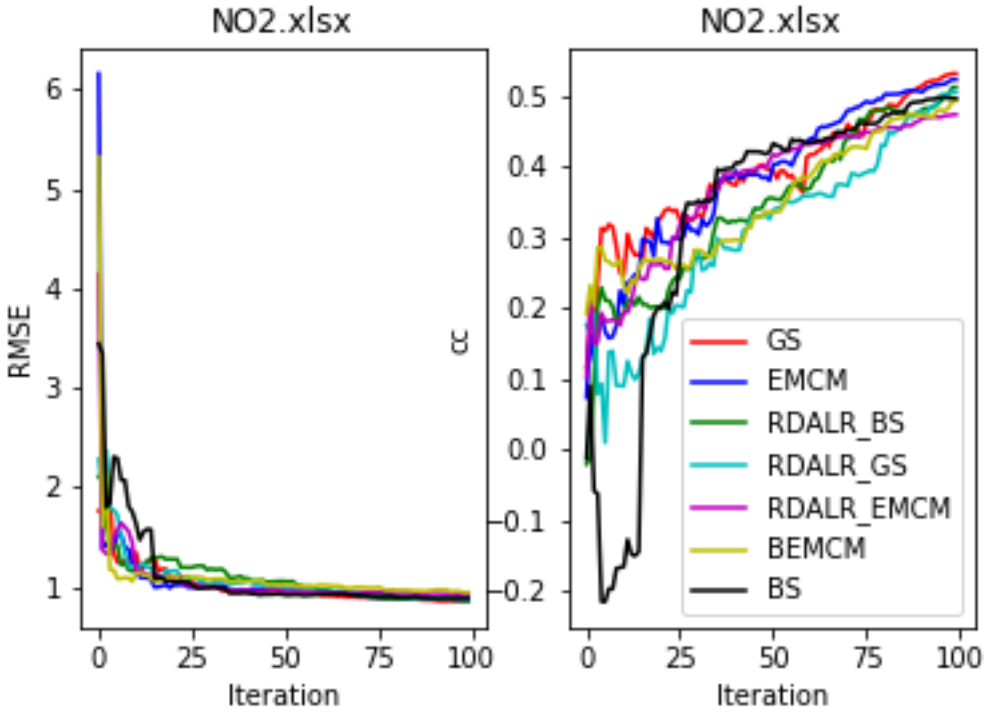

Figura 4.6: RMSE e CC de N02

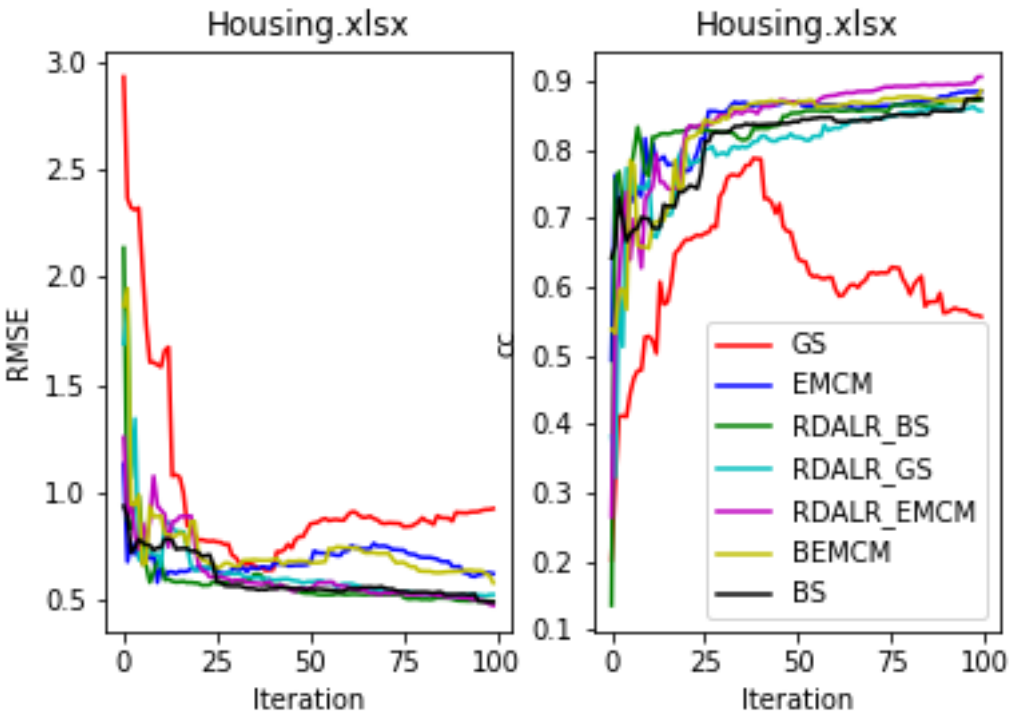

Figura 4.5: RMSE e CC de Housing

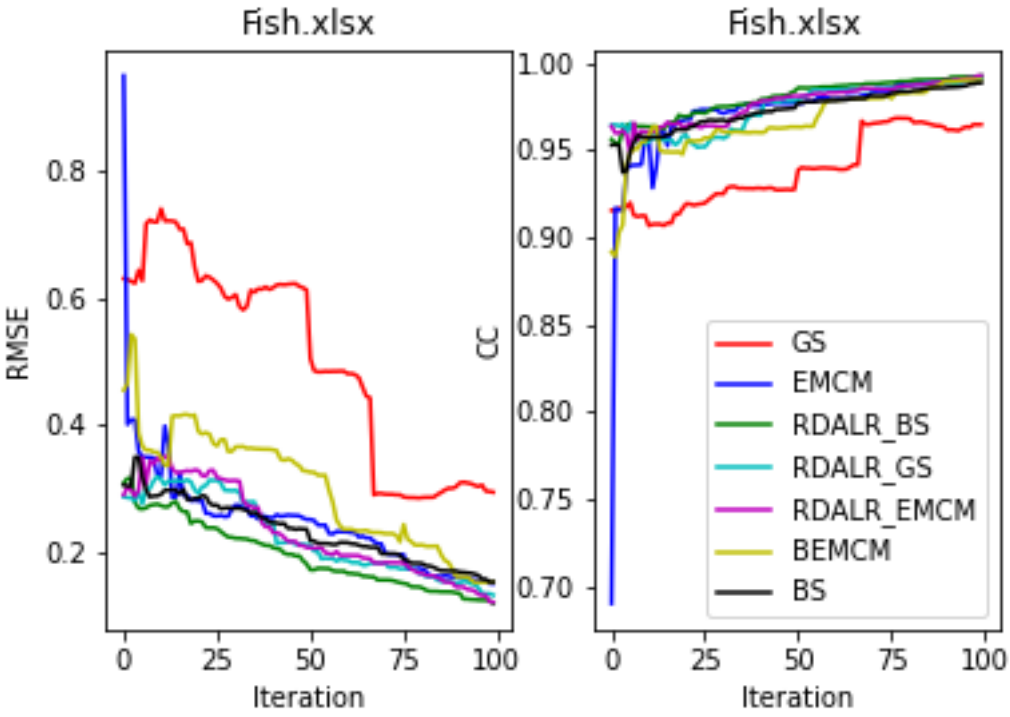

Figura 4.10: RMSE e CC de Fish

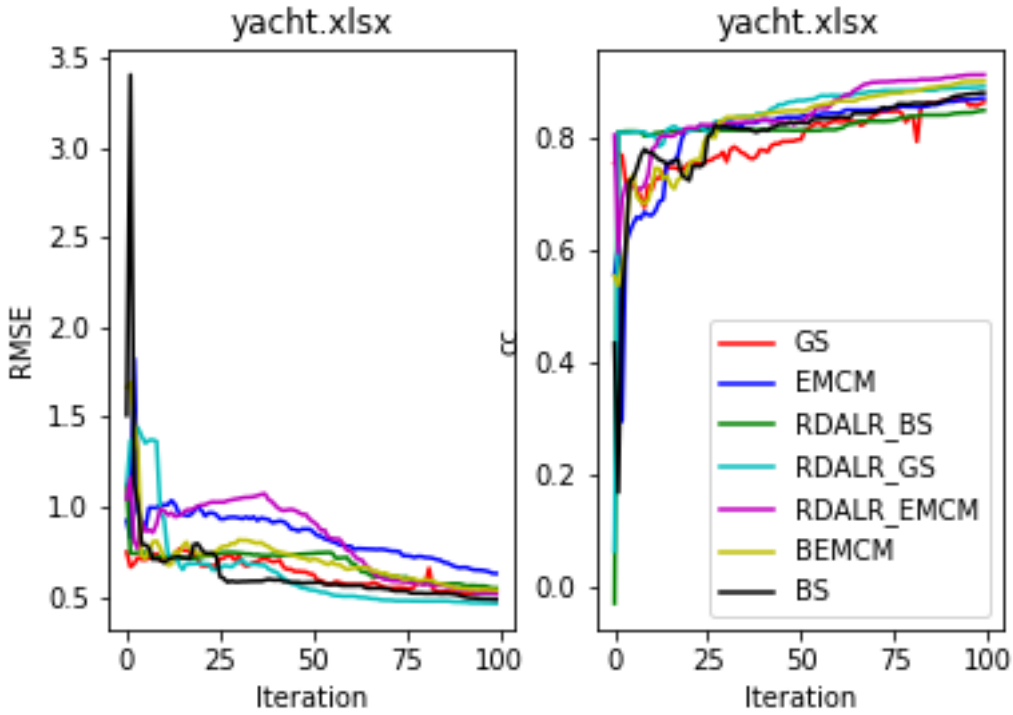

Figura 4.7: RMSE e CC de yacht

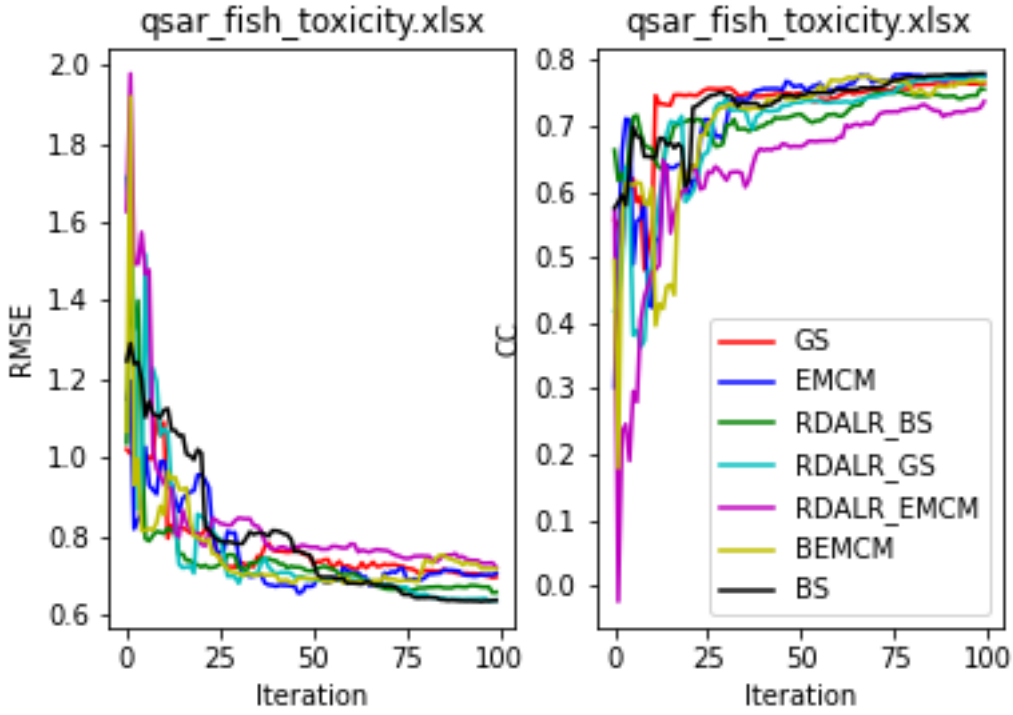

Figura 4.11: RMSE e CC de qsar-fish-toxicity

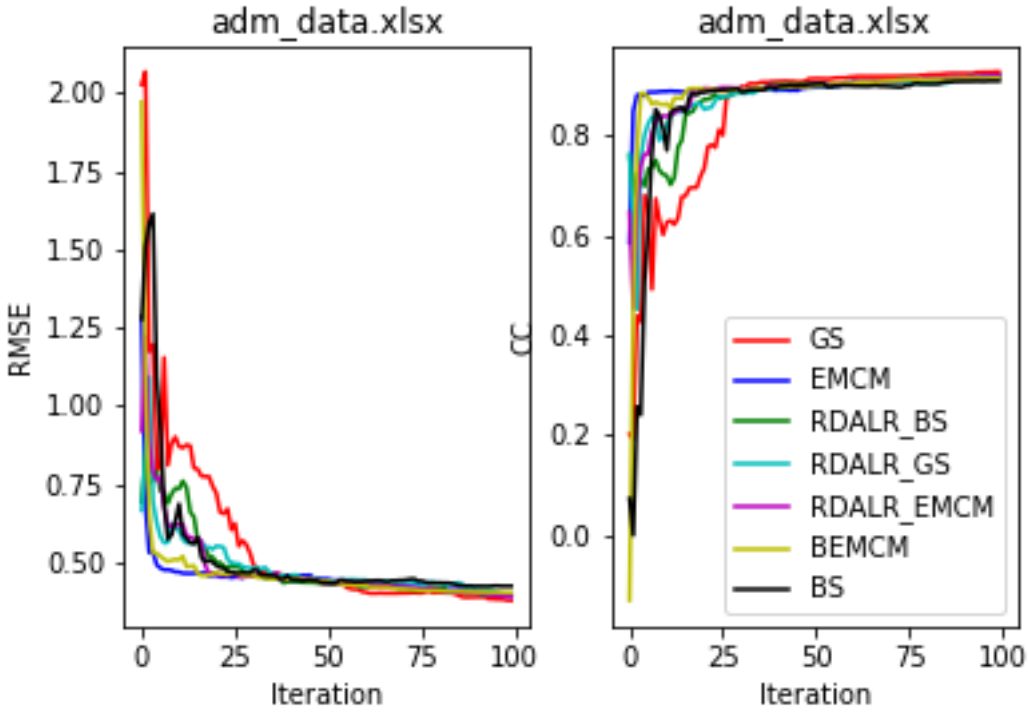

Figura 4.8: RMSE e CC de adm-data

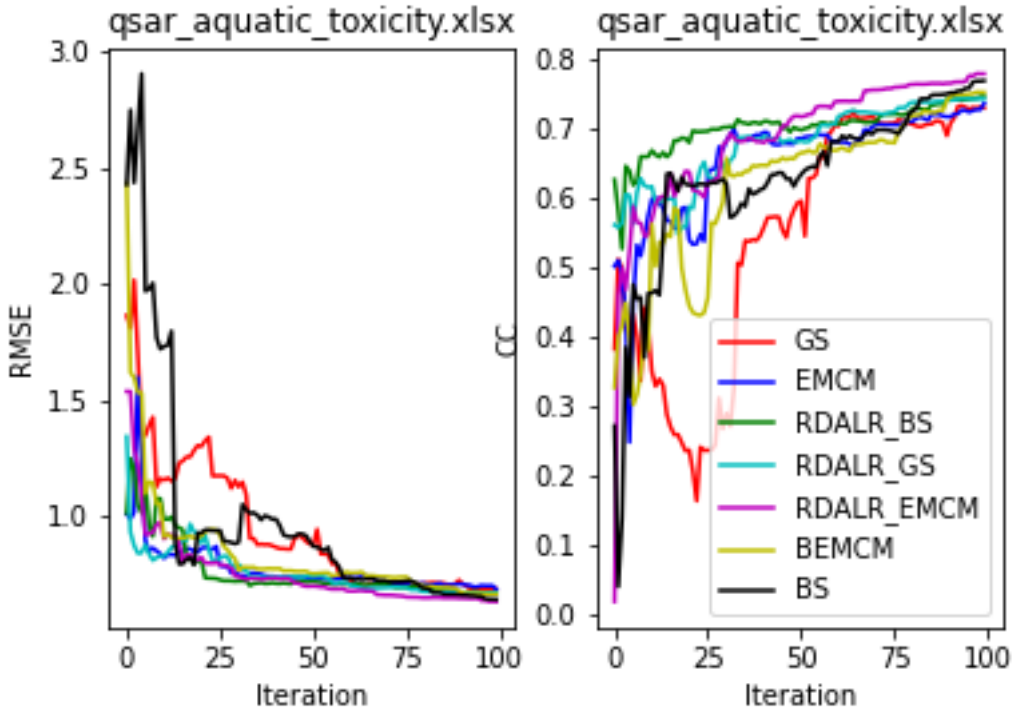

Figura 4.12: RMSE e CC de qsar-aquatic-toxicity

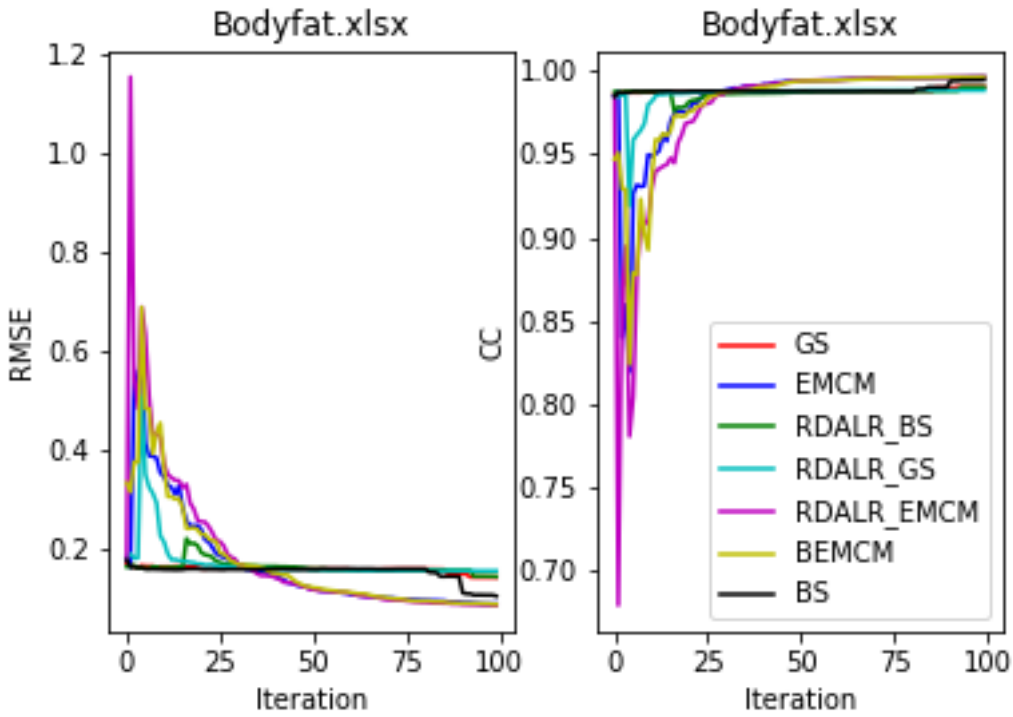

Figura 4.9: RMSE e CC de Bodyfat

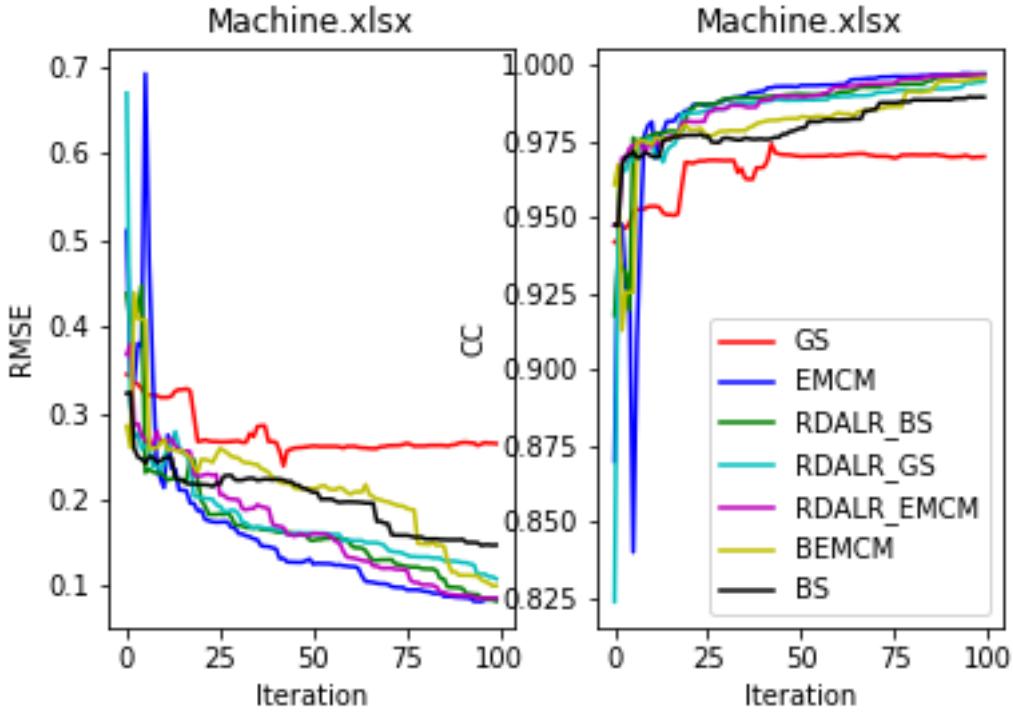

Figura 4.13: RMSE e CC de Machine

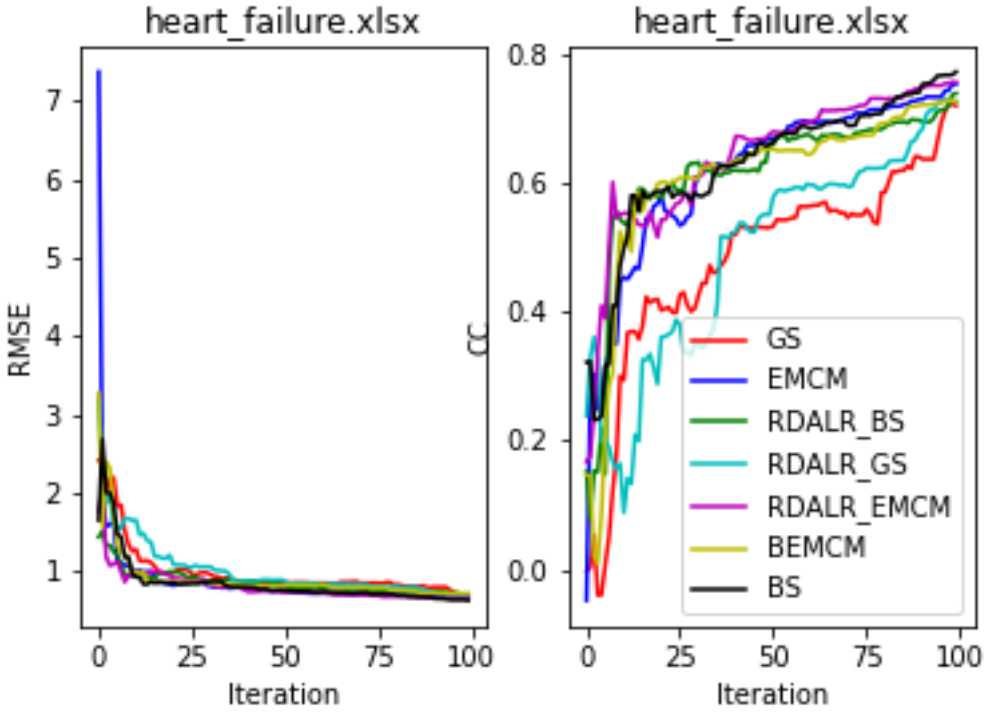

Figura 4.14: RMSE e CC de heart-failure

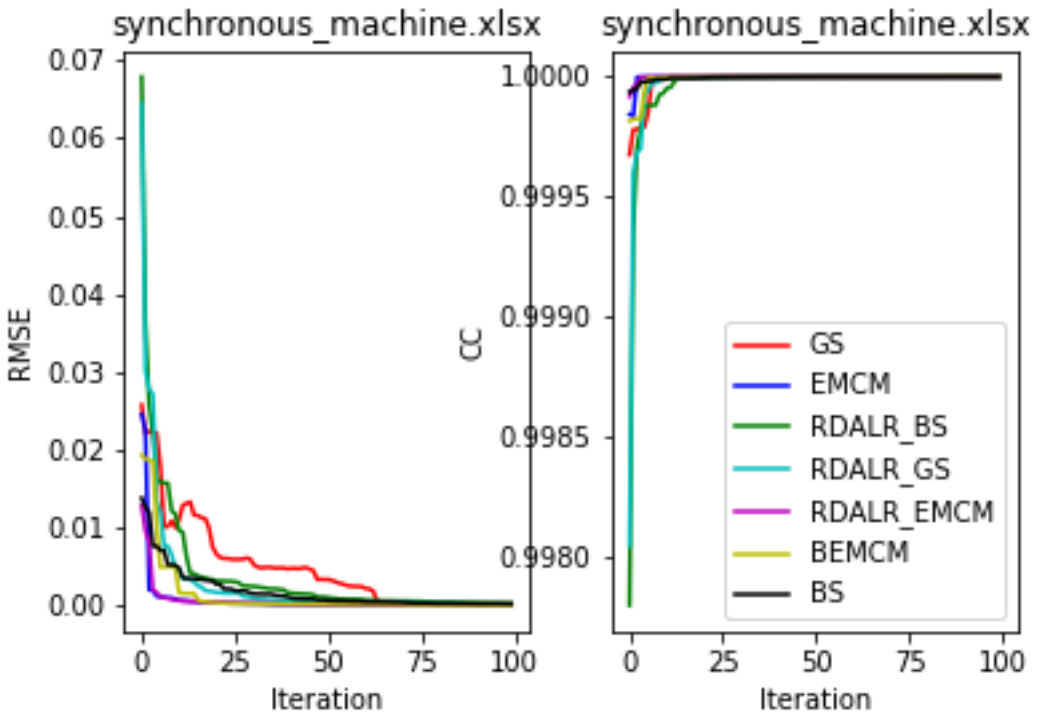

Figura 4.15: RMSE e CC de synchronous-machine

Abaixo está uma tabela que mostra as posições dos algoritmos baseado nas métricas utilizadas. Sendo assim, o algoritmo que alcançar menor RMSE, logo o que teve melhor performance, estará na posição 1, enquanto o algoritmo que alcançar menor CC implicará na posição mais alta.

Rank de performance entre os algoritmos apresentados								
Métrica	Dataset	GS	EMCM	RDALR-BS	RDALR-GS	RDALR-EMCM	BEMCM	BS
RMSE	autoMPG	7	4	1	6	3	5	2
	Concrete	7	6	3	4	1	5	2
	Housing	7	5	3	4	2	6	1
	NO2	5	4	3	1	7	6	2
	yacht	6	7	5	1	2	4	3
	adm-data	1	5	4	3	2	6	7
	Bodyfat	5	3	6	7	1	2	4
	Fish	7	4	1	3	2	5	6
	qsar-fish-toxicity	4	5	3	2	7	6	1
	qsar-aquatic-toxicity	7	6	4	5	1	3	2
	Machine	7	2	1	5	3	4	6
	heart-failure	6	3	4	5	2	7	1
	synchronous-machine	6	1	7	4	3	2	5
CC	autoMPG	7	2	6	3	5	1	4
	Concrete	7	3	4	6	1	2	5
	Housing	7	2	5	6	1	4	3
	NO2	6	1	3	4	7	5	2
	yacht	6	5	7	3	1	2	4
	adm-data	1	5	4	3	2	6	7
	Bodyfat	5	3	6	7	1	2	4
	Fish	7	4	1	3	2	5	6
	qsar-fish-toxicity	5	2	6	3	7	4	1
	qsar-aquatic-toxicity	7	6	4	5	1	3	2
	Machine	7	2	1	5	3	4	6
	heart-failure	7	3	4	6	2	5	1
	synchronous-machine	6	1	7	4	3	2	5

4.6 Análise

Através dos gráficos presentes nos resultados foi possível observar que, no geral, os valores de RMSE tenderam à diminuir, enquanto os valores de CC aumentaram, sugerindo que com o passar das iterações os modelos de regressão alcançaram uma melhor performance, o que é de se esperar.

Observando a tabela acima, outro fator à ser analisado foi que o algoritmo de baseline, que escolhe uma amostra randomicamente à cada iteração, somente performou pior que o algoritmo de RDALR-EMCM. Esse resultado não era esperado uma vez que em (Dong 2019) e (Wenb 2017) o algoritmo baseline teve um desempenho pior na maioria dos datasets. Uma possível causa disso pode ser o fato do algoritmo baseline ter feito boas escolhas de amostras.

Em (Dong 2019) as integrações RDALR-EMCM e RDALR-GS superam tanto o algoritmo RDALR-BS como suas respectivas versões mais simples (EMCM e GS). Segundo a tabela de performance o algoritmo RDALR-EMCM também teve esse comportamento, porém o algoritmo RDALR-GS não superou o RDALR-BS.

Todas as versões do algoritmo RDALR superaram os demais algoritmos na maioria dos datasets.

Ainda, o algoritmo que alcançou a melhor performance baseada nas duas métricas de avaliação foi o algoritmo RDALR-EMCM. Esse resultado também foi visto no estudo feito em (Dong 2019). Ele performou melhor que todos os outros algoritmos em 3 datasets e ficou em segundo em 5.

Já o algoritmo que teve a pior performance entre os demais analisados foi o GS. Ele foi o que performou pior na maioria dos datasets utilizados. O que não era esperado, já que ele superou o EMCM em (Dong 2019).

Além disso é possível comparar os algoritmos de *Expected Model Change Maximization* e sua versão em modo *batch*, *Batch-Mode Expected Model Change Maximization*. A tabela mostra que a versão em *batch* teve uma performance inferior, porém próxima, ao modo sequencial, o que é de se esperar uma vez que no modo sequencial os parâmetros do modelo de regressão são atualizados à cada iteração. Esse resultado também foi observado em (Wenb 2017).

5
Conclusões

Aprendizado ativo é frequentemente utilizado em cenários em que anotação de dados seja custosa, em particular para tarefas de regressão. Isso só é possível através de algoritmos que visam escolher o melhor dado para ser anotado, de tal maneira que o modelo de regressão possua uma boa performance utilizando uma base de dados reduzidas.

Com esse estudo foi possível aprender sobre diferentes técnicas de aprendizado ativo para regressão presentes em (Wenb 2017) e (Dong 2019). Além disso foram confirmados os resultados em (Dong 2019), uma vez que a técnica de integração do RDALR com outros algoritmos obteve resultados melhores do que os algoritmos puros. E o fato do *batch-mode* ter tido uma pior performance que sua variante *sequential-mode* também atendeu as expectativas como apresentado em (Wenb 2017).

Uma possibilidade de trabalho futuro é investigar a integração do algoritmo de RDALR com os algoritmos em *batch-mode* de active learning para que gere resultados melhores dos obtidos em (Dong 2019).

Referências bibliográficas

[GS] YU, H.; KIM, S.. **Passive sampling for regression**. In: 2010 IEEE INTERNATIONAL CONFERENCE ON DATA MINING, p. 1151–1156. IEEE, 2010.

[Beale Regression] BEALE, C. M.; LENNON, J. J.; YEARSLEY, J. M.; BREWER, M. J. ; ELSTON, D. A.. **Regression analysis of spatial data**. Ecology letters, 13(2):246–264, 2010.

[DEAP 2019] KOELSTRA, S.; MUHL, C.; SOLEYMANI, M.; LEE, J.-S.; YAZDANI, A.; EBRAHIMI, T.; PUN, T.; NIJHOLT, A. ; PATRAS, I.. **Deap: A database for emotion analysis using physiological signals**. IEEE Trans. on Affective Computing, 3(1):18–31, 2019.

[Dong 2019] WU, D.. **Pool-based sequential active learning for regression**. IEEE Transactions on Neural Networks and Learning Systems, 30(5):1348–1359, Maio 2019.

[EMCM] CAI, W.; ZHANG, Y. ; ZHOU, J.. **Maximizing expected model change for active learning in regression**. In: 2013 IEEE 13TH INTERNATIONAL CONFERENCE ON DATA MINING, p. 51–60. IEEE, 2013.

[Wenb 2017] CAI, W.. **Batch mode active learning for regression with expected model change**. IEEE Transactions on Neural Networks and Learning Systems, 28(7):1668–1681, Julho 2017.

[gross2003linear] GROSS, J.; GROSS, J.. **Linear regression**, volumen 175. Springer Science & Business Media, 2003.

[settles2009active] SETTLES, B.. **Active learning literature survey**. 2009.