Desenvolvimento de modelos de Machine Learning em Haskell (KNN, Árvore de Decisão e Naive Bayes Gaussiano)

NOME	RA	TURMA
João Pedro Sousa Bianchim	11201920729	**DA1**
João Vitor Arruda de Bartolo	11201812168	**DA1**
Pedro Paulo Ayala Yamada	11202020731	**DA2**

Metodologia: O projeto procura criar algoritmos de Machine Learning em Haskell para solucionar a classificação de câncer de mama (B: Benigno e M: Maligno), os dados foram obtidos via Kaggle, no link aqui.

Inicialmente, foi realizado um benchmarking em *Python*, utilizando a biblioteca *SKlearn*, os resultados serão apresentados na tabela de resultados. Então, os alunos escolheram, cada um, um dos três modelos para o desenvolvimento, a divisão feita foi a seguinte:

- João Arruda: KNN + Métricas de desempenho + Leitura de dados.
- João Bianchim: Naive Bayes Gaussiano.
- Pedro Paulo: Árvore de Decisão.

Como rodar os códigos: Caso queira rodar o benchmarking em Python, basta instalar o *Jupyter Notebook* e as bibliotecas *Sklearn* e *Pandas*. O código utilizado está na pasta *Python Benchmark,* cujo arquivo *Jupyter Notebook* está com o nome *benchmarks.ipynb.*

No projeto desenvolvido em Haskell, é necessário instalar o cassava com o comando:

```
cabal install --lib cassava
```

Então, basta abrir o projeto, contido na pasta projeto-ml e rodar o comando: `stack run`

Destaques: Dentre os destaques em relação às dificuldades do projeto, temos que, dado que o Haskell é uma linguagem puramente funcional, tivemos diversas dificuldades para o debbuging e desenvolvimento dos algoritmos, visto que estávamos acostumados com linguagens imperativas. Para o algoritmo KNN, podemos destacar a composição de funções utilizadas para calcular as distâncias de maneira limpa.

Para o algoritmo de árvores de decisão, podemos destacar a criação do tipo de dados recursivo: ArvoreClassificação, que tem os construtores Classe (que armazena String de classe) ou Decisao (que armazena uma função lambda, que dado um ponto no dataset retorna outras Árvores de Decisão).

Por último para o Naive Bayes, podemos destacar a criação de um modelo (classe) que armazena as médias e desvios padrões de cada classe da classificação e a predição, feita a partir de três outras funções, para possibilitar a análise de tudo armazenado no modelo treinado.

Métricas	KNN Python	Árvore Python	Naive Bayes Python	KNN	Árvore	Naive Bayes
Acurácia	0.9590	0.9298	0.9415	0.9590	0.9298	0.7602
Recall	0.9827	0.8805	0.9344	0.9827	0.9047	0.6057
Precisão	0.9047	0.9365	0.9047	0.9047	0.9047	1.0
F1 Score	0.9421	0.9076	0.9193	0.9421	0.9047	0.7544

Link para o vídeo: https://www.youtube.com/watch?v=qw6eIlpPT8o