# Análise de Algoritmos de Aprendizado de Máquina Em Um Problema de Classificação

**Matheus E. Santana**[1], **Pedro H. Fukuda**[2]

[1]Faculdade de Computação – Universidade Federal do Mato Grosso do Sul (UFMS)
Campo Grande – MS – Brasil

matheusess@hotmail.com, pedrohfukuda@gmail.com

***Abstract.*** *This paper describes the findings of a comparative analysis of 5 (five) choosen machine learning algorithms to solve the classification problem Ghouls, Goblins, and Ghosts... Boo! from the Kaggle plataform (kaggle.com). The choosen algorithms are: KNN - K-Nearest-Neighbors, SVM - Support Vector Machine, Naive-Bayes, Logistic Regression and Decision Tree. As classification is the nature of the choosen problem, the analysis consists of metrics to estimate the accuracy of each algorithm and the final results of such comparisons as they are submitted to the Kaggle plataform.*

***Resumo.*** *Este artigo descreve os resultados das análises comparativas de cinco algoritmos de aprendizado de máquina, escolhidos para resolver o problema de classificação Ghouls, Goblins, and Ghosts... Boo! da plataforma kaggle.com. Os algoritmos são: KNN - K-Nearest-Neighbors, SVM - Support Vector Machine, Naive-Bayes, Regressão Logística e Árvore de Decisão. Sendo classificação a natureza do problema proposto, as análises consistem em métricas para estimar a acurácia obtida de cada algoritmo, e o seus resultados finais ao serem submetidos na plataforma kaggle.*

## 1. Ghouls, Goblins, and Ghosts... Boo! - Descrição

Este problema, disponibilizado pela plataforma kaggle na categoria Playground, se refere a classificação de monstros em três classe: ghouls, globins e ghosts. Tendo para todos eles as seguintes características:

1. ***id*** - número de identificação,
2. ***bone length*** - comprimento médio do osso,
3. ***rottingflesh*** - porcentagem de podridão,
4. ***hair length*** - comprimento médio do cabelo,
5. ***has soul*** - porcentagem de alma, ***color*** - cor dominante
6. ***type*** indicando a classe que o monstro pertence.

Existem no total, 900 monstros sendo que desta quantidade 371 já foram classificados e todos os seus dados foram disponíbilizados. Este conjunto de dados foi utilizado para o treinamento dos algoritmos, para a classificação dos demais monstros.

## 2. Ferramentas Utilizadas

Para o estudo das diferentes técnicas foi escolhida a biblioteca de machine-learning *Scikit-learn* versão 0.19.1, utilizada na linguagem *Python* versão 3.6.3, em conjunto das bibliotecas *Pandas* na sua versão 0.23.0, e *Numpy* na versão 1.14.3, para a manipulação dos dados. Além disso, fizemos o uso da biblioteca *Seaborn*, para a vizualização dos gráficos referentes as características do conjunto de treinamento.

## 3. Descrição dos Atributos

Em sequência descrevemos com detalhes cada atributo:

1. Id: Atributo numérico referente a identificação do monstro.
2. Bone Length: Atributo numérico normalizado entre 0 e 1.
3. Rotting Flesh: Atributo numérico normalizado entre 0 e 1.
4. Hair Length: Atributo numérico normalizado entre 0 e 1.
5. Has Soul: Atributo numérico normalizado entre 0 e 1.
6. Color: Atributo categórico podendo ser: white, black, clear, blue, green, blood.
7. Type - Atributo categórico podendo ser: Ghost, Goblin e Ghoul. Sendo este atributo alvo da classificação.

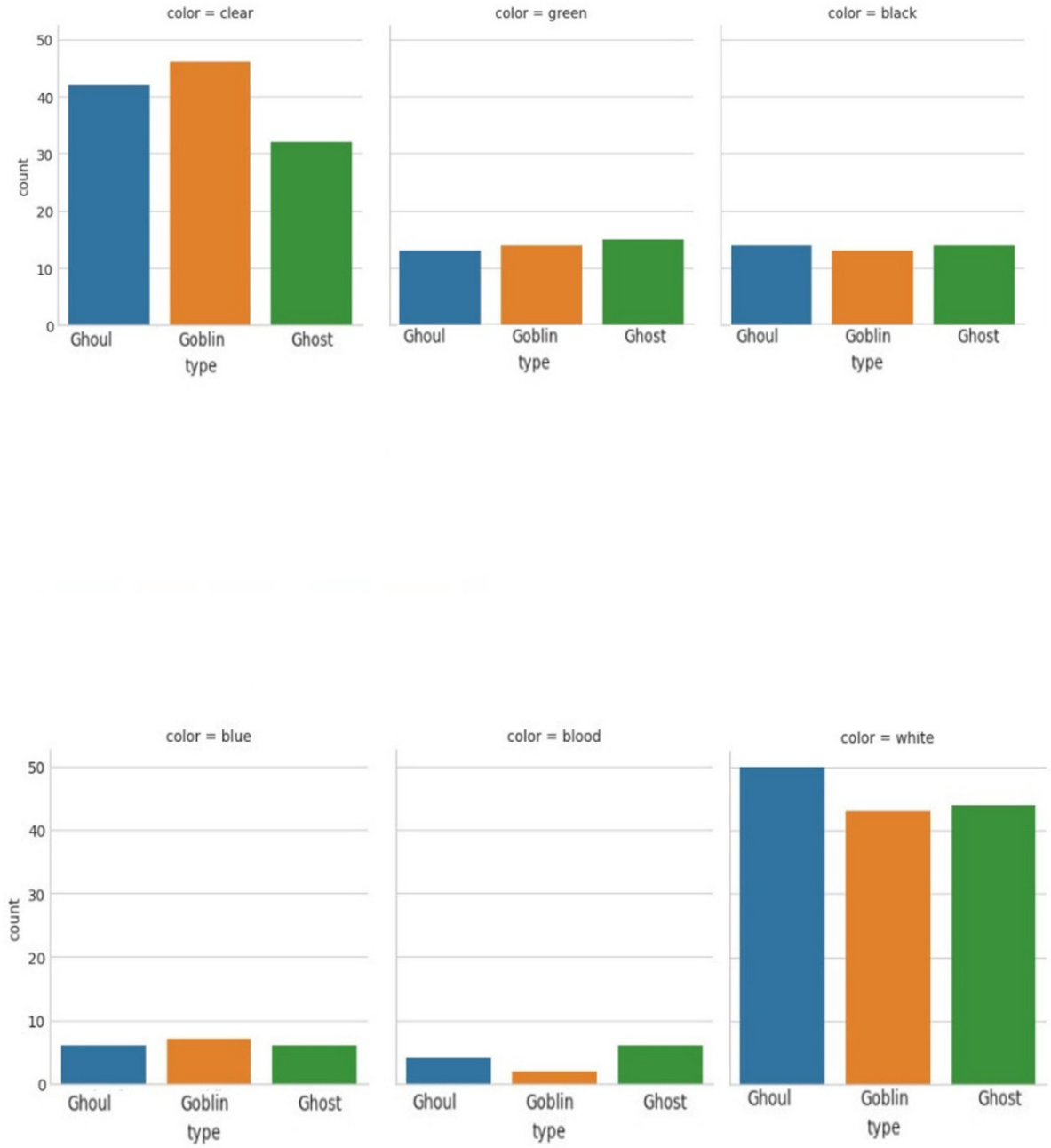
**Amostra de Dados de Treinamento**

| | id | bone_length | rotting_flesh | hair_length | has_soul | color | type |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0.354512 | 0.350839 | 0.465761 | 0.781142 | clear | Ghoul |
| 1 | 1 | 0.575560 | 0.425868 | 0.531401 | 0.439899 | green | Goblin |
| 2 | 2 | 0.467875 | 0.354330 | 0.811616 | 0.791225 | black | Ghoul |
| 3 | 4 | 0.776652 | 0.508723 | 0.636766 | 0.884464 | black | Ghoul |
| 4 | 5 | 0.566117 | 0.875862 | 0.418594 | 0.636438 | green | Ghost |
| 5 | 7 | 0.405680 | 0.253277 | 0.441420 | 0.280324 | green | Goblin |
| 6 | 8 | 0.399331 | 0.568952 | 0.618391 | 0.467901 | white | Goblin |
| 7 | 11 | 0.516224 | 0.536429 | 0.612776 | 0.468048 | clear | Ghoul |
| 8 | 12 | 0.314295 | 0.671280 | 0.417267 | 0.227548 | blue | Ghost |
| 9 | 19 | 0.280942 | 0.701457 | 0.179633 | 0.141183 | white | Ghost |

## 4. Manipulando Atributo Categórico

Um dos atributos do conjunto de dados é o atributo *color*. Sendo este atributo categórico, dado que todos os outros atributos são númericos, foi necessário desenvolver uma métrica para manipulá-lo de forma a transformá-lo em um atributo numérico também, para obter um melhor desempenho nos algoritmos de aprendizado baseados em modelos matemáticos. Sendo assim ao buscar-mos uma solução para a normalização desta característica, analisando todo o conjunto de dados levando em consideração esse atributo em particular, notamos que a diferença média de cores entre os diferentes tipos de monstros, (veja figura 2), é muito pequena. Isto, a princípio, nos mostrou que talvez esse atributo poderia não ter grande relevância para o aprendizado dos algoritmos.

**figura 2: Média de Cores**




Portanto executamos todos os algoritmos incluindo e excluindo esse atributo. E assim conseguimos algumas melhoras significativas na precisão dos modelos de classificação, (veja a figura 3).

| Algoritmo | Taxa de Acerto sem cor | Taxa de Acerto com Cor | Ganho |
|---|---|---|---|
| Knn | 0.76 | 0.8666666 | +0.106 |
| Svm | 0.813333 | 0.84 | +0.27 |
| Regressão Logística | 0.82666666 | 0.8533333 | +0.027 |
| Naive Bayes | 0.76 | 0.84 | +0.08 |
| Árvore de Decisão | 0.5866666 | 0.6533333 | +0.067 |

Figura 3. Ganho de Precisão

## 5. Manipulação do Conjunto de Treinamento

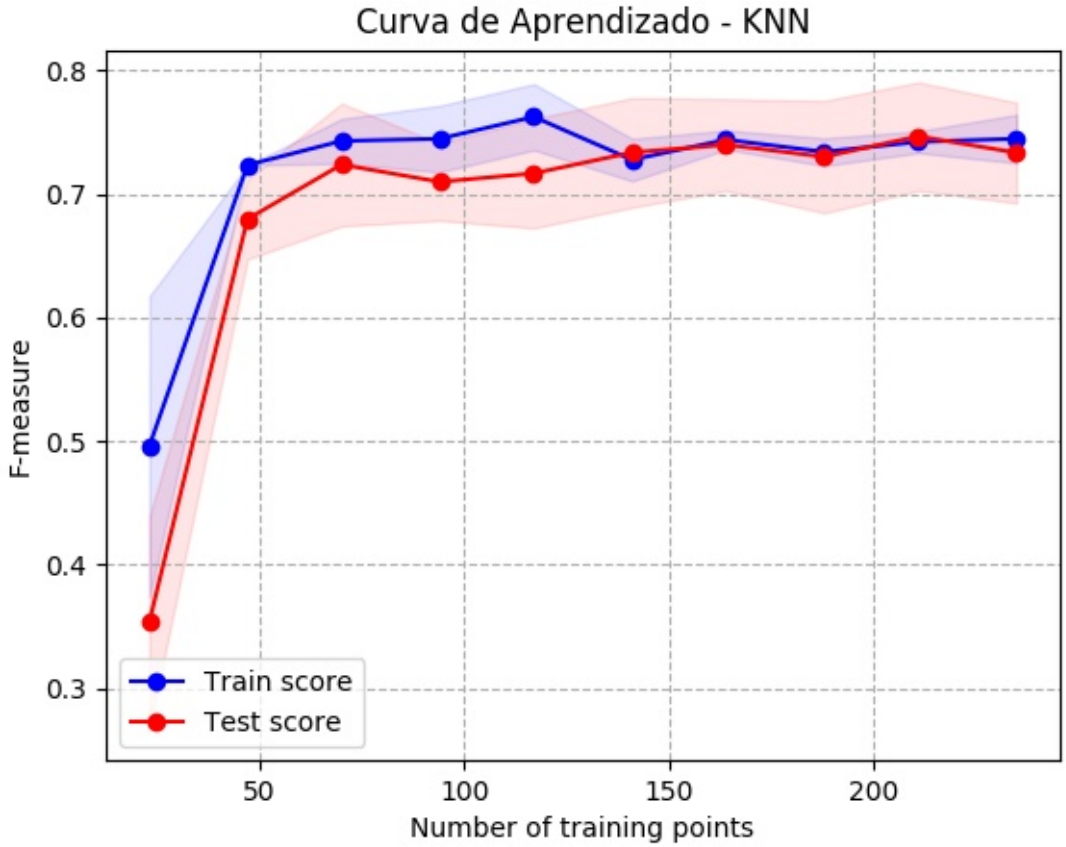
Para apuração dos algoritmos nós dividimos o conjunto de treinamento em 2 partes, utilizando apenas 20% do conjunto para medir a precisão de cada um deles. Os outros 80% do conjunto foram usados para o aprendizado dos classificadores que foram realizado utilizando a métrica de k-folds com $k = 5$. Onde a cada iteração dos algoritmos nós embaralhamos as amostras de forma aleatória e separamos em 5 folds de tamanhos iguais, usando 4 deles para aprendizado e 1 como conjunto de validação. Utilizamos a ferramenta GridSearchCV da biblioteca Scikit-Learn para a separar o melhor classificador e os melhores parâmetros para ele. Dado o melhor classificador nós verificamos a sua taxa de acerto no conjunto de teste.

## 6. Hiperparâmetros e Curvas de Aprendizado

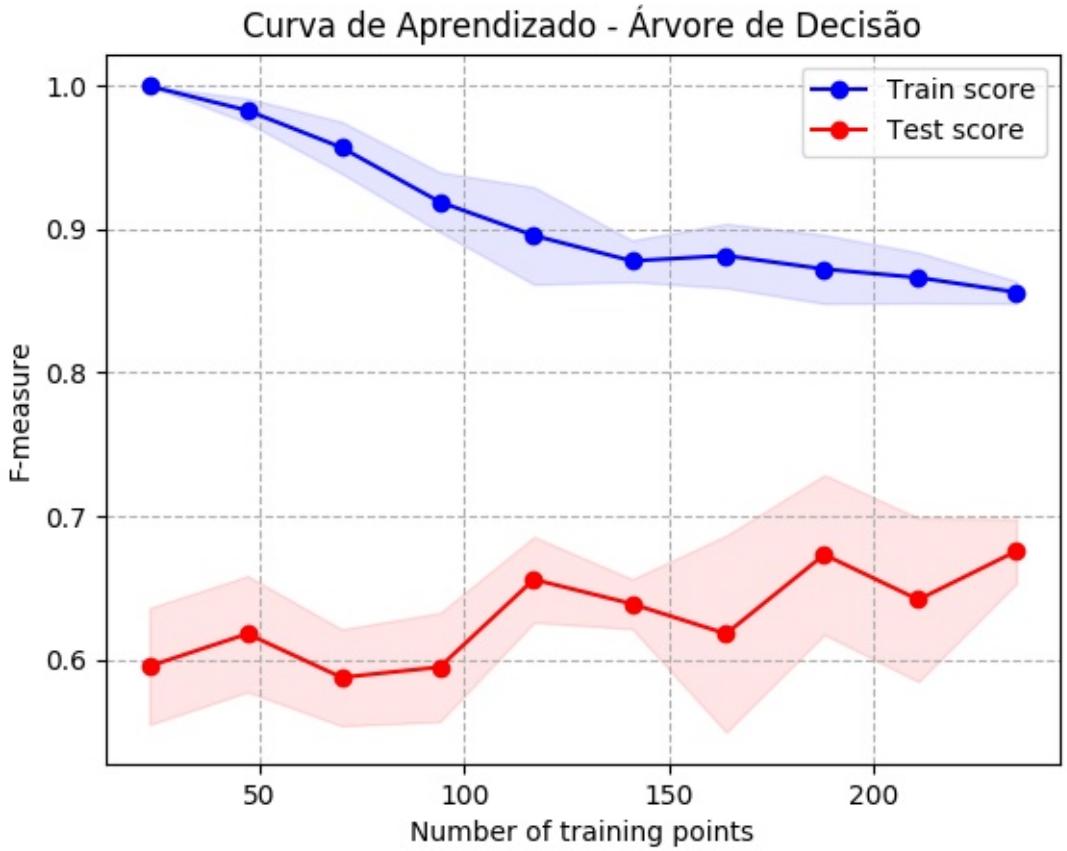
Nesta seção vamos descrever detalhes sobre a média de acerto, parâmetros e hiperparâmetros de cada modelo.

1. K-Nearest-Neighbors

   Para este modelo nós escolhemos 5 valores para K, sendo eles $(3, 5, 10, 15, 20)$, referente ao número de vizinhos que o algoritmo selecionará para classificar a amostra dada, e duas métricas de ponderação, a ponderação uniforme, na qual todos as amostras em uma vizinha são ponderadas de forma igual, e a ponderação por distância onde o peso é dado de forma inversa a distância da amostra aos vizinhos mais próximos. Os melhores parâmetros retornados pelo GridSearch foram: $K = 20$ e a ponderação uniforme. Este parâmetros forneceram uma taxa de acerto de 73% no conjunto de validação. A seguir o gráfico representa a curva de aprendizado deste modelo.

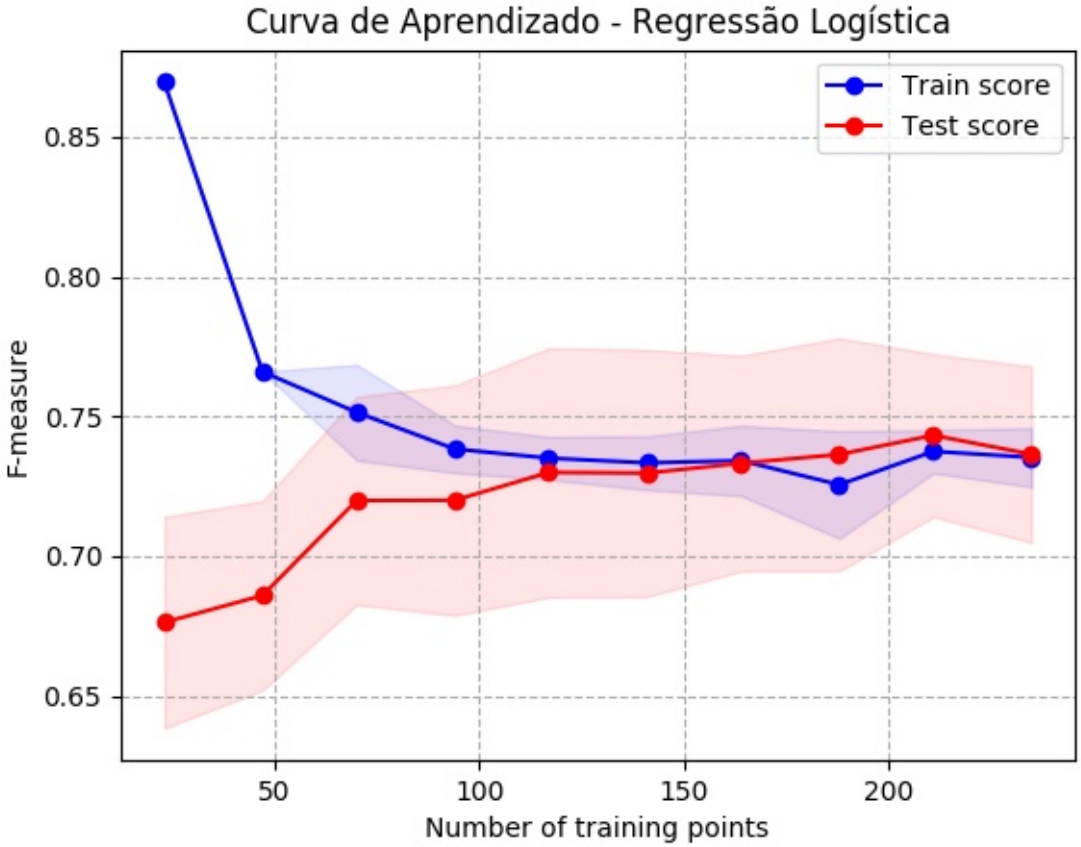



2. Árvore de Decisão
Neste modelo utilizamos o parâmetro de Max-Depth com os valores $(1, 2, 3, 4, 5)$ referente as profundidades das diferentes topologias das árvores geradas,e o parâmetro Max-Feature referente ao número de características que serão considerados quando o algoritmo determinar a melhor árvore. Para o Max-Feature usamos os valores $(1, 2, 3, 4)$. A taxa de acerto do conjunto de validação deste modelo foi de $0.6722$, e os melhores parâmetros encontrados foram: Max-Depth = 3, Max-Features=4. A seguir o gráfico representa a curva de aprendizado deste modelo.

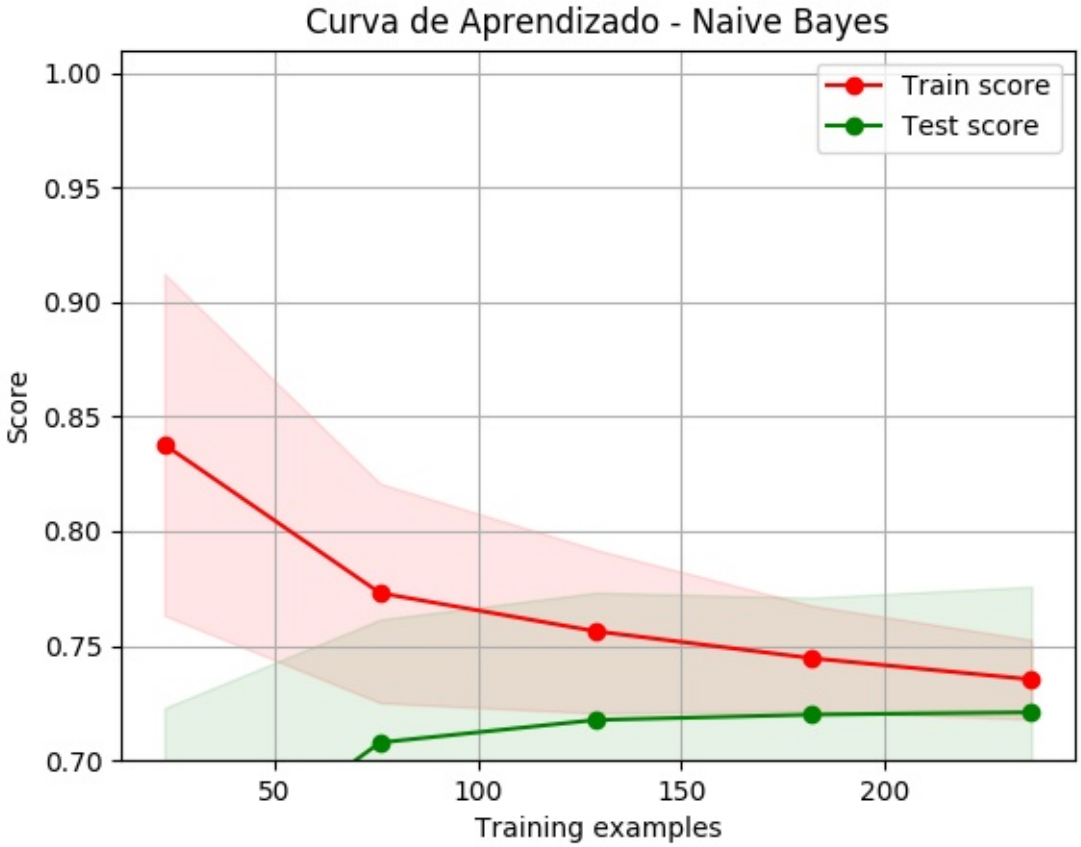



3. Regressão Logística

Para esta técnica nós utilizamos apenas um único parâmetro $C$ referente ao inverso da regularização. Os valores escolhidos para este parâmetro foram $(0.005, 0.01, 0.05, 1, 2.5, 5, 10, 100, 1000, 10000000)$, sendo $10000$ o melhor valor encontrado fornecendo $73\%$ de acerto no conjunto de validação. A seguir o gráfico representa a curva de aprendizado deste modelo.




4. SVM - Support Vector Machine.
Este modelo apresentou o melhor resultado dentre todos os algoritmos. A sua porcetagem de acerto no conjunto de validação é foi de $74\%$. Os parâmetros escolhidos para este algoritmos foram linear e rbf como métricas de kernel, valores $(1, 3, 5, 10)$ como valores inversos a regularização, e os valores $(3, 5, 10)$ para os graus dos polinômios. Os melhores parâmetros encontrados foram $C = 1$, grau de polinômio = 3, e kernel linear. A seguir o gráfico representa a curva de aprendizado deste modelo.




5. Naive Bayes.
Por fim, este modelo também apresentou uma boa precisão na sua classificação. Para este nós não usamos hiper-parâmetros e mesmo assim ele foi o segundo melhor modelo que obtivemos, apresentando uma média de acertos aproximada de 73%. A seguir o gráfico representa a curva de aprendizado deste modelo.

## 7. Média Acertos e Acurácia

O gráfico abaixo mostra a média de acertos de cada classificador para cada tipo de monstro, e a acurácia obtida no conjunto de teste.

| Algoritmo | | KNN | Árvore de decisão | Regressão logística | SVM | Naive Bayes |
|---|---|---|---|---|---|---|
| Precisões | Ghost | 0.86 | 0.63 | 1.00 | 0.96 | 0.96 |
| | Ghoul | 0.88 | 0.69 | 0.88 | 0.88 | 0.85 |
| | Goblin | 0.86 | 0.62 | 0.67 | 0.67 | 0.71 |
| Média | | 0.87 | 0.65 | 0.88 | 0.85 | 0.85 |
| Acurácia | | 0.866666667 | 0.65333333333333 | 0.853333333333334 | 0.84 | 0.84 |

## 8. Submissão e Considerações Finais

Como podemos observar nos resultados acima, concluímos que para o desafio Ghouls, Goblins, and Ghosts... Boo!, o melhor modelo obtido foi o SVM, tanto na validação cruzada, quanto no conjunto de teste préviamente separado. Em sequência e por poucas

diferenças nós obtivemos os modelos Naive Bayes, o classificador de regressão logística e o K-nearest-Neighbors, que apresentaram bons resultados, estes que inclusive são bastante similares entre sí. Por último tivemos o modelo de árvore de decisão, com um resultado considerávelmente inferior aos demais. Portanto, apesar de termos O SVM com o melhor resultado, nós decidimos submeter na plataforma Kaggle, todos os algoritmos implementados, para por fim obter a os resultados da apuração destes modelos no conjunto de teste real. Tivemos então por conclusão, que a mesma ordem de resultados no conjunto de teste foi a de resultados na plataforma Kaggle (Veja a figura abaixo). Sendo assim escolhemos o SVM como nosso modelo de classificação final.

**Resultado final dos classificadores na plataforma Kaggle.com**