Relatório do Projeto de Aprendizagem de Máquina

Professores:

Cleber Zanchettin

Francisco de Assis Tenório Carvalho

Grupo:

José Antônio da Silva

Pedro Henrique Martins Barbosa

Rodrigo Bruno de Carvalho Cavalcanti

Wilton Pereira Santos Santana

2016.2

Sumário

1. Introdução

O trabalho realizado para a disciplina de Aprendizagem de Máquina implementa o algoritmo de agrupamento VKCM-K [1] e mais quatro técnicas de classificação de dados:

1. Estimação paramétrica por máxima verossimilhança;
2. Rede neural MLP;
3. SVM;
4. Voto Majoritário;

Para avaliar os resultados obtidos e comparar o desempenho das técnicas implementadas, foi utilizada a validação cruzada estratificada, foram obtidos uma estimativa pontual e um intervalo de confiança para a taxa de erro para cada classificador, e foram realizados os testes de Friedman e Nemenyi (pós-teste).

Todas estas técnicas implementadas e de avaliação foram executadas sobre a base de dados Abalone [2], utilizando o software MATLAB. Essa base de dados contém 4177 padrões, cada um com 9 variáveis, sendo uma delas o sexo (M, F ou I) que representa a classe a ser determinada pelos algoritmos.

Este trabalho está organizado da seguinte forma: na Seção 2 será feita uma descrição das técnicas implementadas, a Seção 3 apresentará uma breve descrição das técnicas de avaliação, e na Seção 4 serão mostrados os experimentos realizados e os resultados obtidos; por último, a Seção 5 apresentará algumas considerações sobre o que foi desenvolvido.

2. Técnicas Implementadas

Neste trabalho, quatro técnicas foram implementadas e aplicadas na base de dados *Abalone*, para agrupar ou classificar os dados em três categorias (classes M, F, I). Os algoritmos supervisionados de classificação foram: Classificador Bayesiano com Estimação paramétrica por máxima verossimilhança, rede neural MLP (MultiLayer Perceptron), SVM (Máquina de Vetores de Suporte) e voto majoritário. O algoritmo não-supervisionado de agrupamento desenvolvido foi o VKCM-K.

2.1. VKCM-K

Algoritmos de agrupamento têm o papel de organizar um conjunto de padrões em grupos, de tal modo que os padrões dentro de um mesmo grupo compartilham um elevado grau de semelhança. Um detalhe importante nos algoritmos de agrupamento é a medida de similaridade, as quais, em muitos casos, são realizadas através de medidas de distâncias, como a distância euclidiana. Porém, quando a estrutura de dados é complexa, ou seja, não são linearmente separáveis, esses algoritmos podem apresentar um desempenho insatisfatório.

Para tal problema, alguns métodos já foram propostos, destacando-se os métodos que utilizam a transformação de dados não-lineares para outra dimensão, tornando-os linearmente separáveis, através de funções *kernel*. Ao utilizar esta abordagem, a maioria dos algoritmos de agrupamento existentes, tais como o *K-means* e o SOM, consideram que todas as variáveis dos padrões são igualmente importantes, sendo dado o mesmo peso na construção dos agrupamentos. Porém, quando estamos utilizando dados em alta dimensão, algumas variáveis podem ser

menos importantes que outras para o processo de agrupamento, como também podem ser irrelevantes.

Nesse contexto, o trabalho [1] apresenta o algoritmo VKCM-K, que utiliza o método de agrupamento por kernel no espaço de características, diferenciando-se dos demais algoritmos por apresentar um método para ponderar as variáveis, podendo assim utilizar diferentes pesos e funções kernel para cada variável.

2.2. Estimativa de Máxima Verossimilhança

Para obter um classificador ótimo é necessário conhecer previamente as probabilidades a *priori* e as densidades condicionais das classes. Porém, em casos reais, essas informações geralmente não estão disponíveis, tendo apenas um conhecimento muito geral da situação e a amostra dos exemplos a serem classificados.

Nesses casos, para simplificar o problema é necessário supor conhecida a forma paramétrica da função densidade: $p(\overline{x}\,|\omega_i) \sim N(\mu_i, \Sigma_i)$, onde os parâmetros μ_i e Σ_i são desconhecidos. Com isso, ao invés de estimar a função densidade $p(\overline{x}\,|\omega_i)$, estimamos os parâmetros μ_i e Σ_i, utilizando uma abordagem de estimação de parâmetros, que pode ser a máxima verossimilhança, a estimação bayesiana ou a densidade de misturas. Neste projeto, apenas a estimação paramétrica por máxima verossimilhança foi utilizada. Nela, para estimar os parâmetros são feitas as seguintes suposições:

- $p(\overline{x}\,|\omega_i)$ possui uma forma paramétrica conhecida determinada pelo valor do vetor de parâmetros θ_i ;
- $p(\overline{x}\,|\omega_i) \sim N(\mu_i, \Sigma_i)$;
- $\overline{\theta_i} = (\mu_i, \Sigma_i)$ então $p(\overline{x}\,|\omega_i) \sim N\left(\overline{\theta_i}\right) \Rightarrow p\left(\overline{x}\,|\omega_i, \overline{\theta_i}\right)$;
  Os parâmetros de diferentes classes são independentes;

Através dessas suposições, para estimar $\overline{\theta_i}$ basta utilizar o conjunto de treinamento, que tem seus exemplos construídos independentemente a partir da

função densidade de probabilidade $p(\overline{x}|\omega_i)$. Assim, a estimativa de máxima verossimilhança de θ i é o valor de θ i que maximiza $p\left(D \mid \overline{\theta}_i\right)$.

2.3. MLP

MLP é uma rede neural artificial do tipo perceptron que contém uma camada de entrada, pelo menos uma camada escondida (intermediária) e uma camada de saída. Na camada intermediária cada neurônio escondido possui uma função de ativação para receber a ponderação entre a camada de entrada e os pesos de cada conexão e propagar uma saída dentro do intervalo de sua função. Na camada de saída os neurônios recebem os valores da camada intermediária e aplicam a eles uma função de saída, retornando o valor de saída da rede.

O treinamento das redes MLP geralmente ocorre através do algoritmo supervisionado chamado *backpropagation*, que contém duas etapas. Na primeira, um padrão é apresentado à camada de entrada da rede neural. Em seguida, os neurônios desta camada computam suas respostas, que servirão de entradas para a camada posterior, até que na camada de saída o valor do erro é calculado. Na segunda etapa, o erro é propagado da camada de saída até a camada de entrada, e os pesos das conexões dos neurônios vão sendo atualizados de acordo com a regra delta generalizada.

Ao final do treinamento da rede, por exemplo quando o erro alcançar um nível mínimo satisfatório, a rede pode ser usada para classificação de novos padrões. Esta fase de teste é completamente progressiva (*feedforward*), já que os padrões são apresentados à camada de entrada, processados nas camadas intermediárias e os resultados são apresentados na camada de saída, sem que haja a retropropagação (*feedbackward*) do valor do erro.

Dentre as principais limitações das redes neurais que utilizam o backpropagation como algoritmo de treinamento, destaca-se o fato de ela serem vistas como "caixas pretas", já que não se sabe ao certo como nem por que a rede chegou a um determinado resultado, uma vez que não há justificativas para as respostas apresentadas. Além disso, outro ponto a ser destacado é o do tempo de treinamento, que tende a ser muito lento. Algumas vezes demora-se muito para se

chegar a um nível de erro aceitável, por isso são introduzidos outros critérios de parada no treinamento de uma rede neural, tais como erros de validação e número de iterações.

2.4. SVM (Máquina de Vetores de Suporte)

Trata-se de um conceito na área de Ciência da Computação para um conjunto de técnicas de aprendizado supervisionado que analisam os dados, reconhecendo padrões existentes, sendo empregado para classificação de dados (linear e não-linear) e análise de regressão. O SVM utiliza um mapeamento não-linear para transformar os dados originais (treinamento) em uma dimensão maior. Nesta nova dimensão, o SVM busca um hiperplano que gere uma separação linear ótima. Sempre separa duas classes por meio de um hiperplano com a maior distância entre as classes, onde a distância é chamada de margem; o hiperplano é encontrado utilizando vetores de suporte (tuplas especiais na etapa de treinamento) e margens definidas pelos vetores suportes. A SVM busca primeiro classificar as classes, definindo cada ponto pertencente a cada um dos grupos, para em seguida maximizar a margem, ou seja, primeiro é feita a classificação correta para em seguida, em função desta restrição, redefinir a distância entre as margens.

2.5. Voto Majoritário

O Voto Majoritário é uma das técnicas mais utilizadas em Sistemas de Múltiplos Classificadores. Nesta técnica em vez de empregar um único classificador emprega vários classificadores resultando em melhorias consideráveis em relação a sistemas de classificadores singulares.

Há três tipos de classificadores de Voto Majoritário baseado no método em que as decisões são tomadas. O primeiro método é denominado Voto Unânime no qual a classe selecionada é aquela a qual todos os classificadores estão de acordo. O segundo método é chamado de Maioria Simples na qual é uma classe é atribuída se ao menos mais da metade dos classificadores concordam com esta classe. O terceiro é titulado de Votação por Maioria no qual a classe com maior número de

votos é a classe atribuída ao exemplo. Este último é o mais popular dentre os três e sua regra de decisão é:

$$\sum_{i=1}^{N} d_{i,j} = max_{j=1}^{M}(\sum_{i=1}^{N} d_{i,j})$$

No qual N é o número de classificadores combinados, M é o número de classes e $d_{i,j}$ é a decisão do i-ésimo classificador pela j-ésima classe.

No Projeto o método de Votação por Maioria do Voto Majoritário foi adotado sendo combinado o Classificador Bayesiano, MLP e SVM combinados para determinar entre uma das três classes do conjunto de dados Abalone.

3. Técnicas de Avaliação

3.1. Validação Cruzada Estratificada

A validação cruzada é utilizada para verificar a capacidade de generalização de um algoritmo a partir do conjunto de dados que não foi utilizado para o treinamento. Por si só, já torna possível escolher os parâmetros do modelo que apresentam os melhores resultados, economizando tempo de processamento. Para tal, particiona-se o conjunto de dados em grupos mutuamente exclusivos (folds), separando um desses grupos para a validação e o restante para o treinamento. Alterna-se o grupo de validação N vezes, onde N é quantidade de grupos. Dessa forma, o algoritmo pode alterar seus parâmetros e testar em grupos de dados diferentes, podendo apresentar um resultado mais representativo de sua capacidade de generalização. Neste projeto os dados foram divididos em dez folds.

3.2. Estimativa Pontual

A estimativa pontual é o valor obtido por um estimador para uma amostra específica. No projeto a taxa de erro de cada classificador foi utilizada como estimativa pontual.

3.3. Intervalo de Confiança

O intervalo de confiança para um parâmetro θ a um grau de confiança $1-\alpha$ é uma concretização de um intervalo aleatório $\left(L_{inf}, L_{sup}\right)$ de modo que: $P\left(L_{inf} < L_{sup}\right) = 1-\alpha$, onde $\alpha \in (0, 1)$ e deve ser um valor muito pequeno para termos um nível de confiança elevado. O intervalo de confiança ao nível de confiança $(1-\alpha)$ pode ser definido da seguinte forma:

$$\left[\bar{x} - \frac{\sigma}{\sqrt{n}} Z_{\frac{\alpha}{2}}, \bar{x} + \frac{\sigma}{\sqrt{n}} Z_{\frac{\alpha}{2}}\right]$$

Onde $\bar{x}$ é o valor da média da taxa de erro, e $Z_{\frac{\alpha}{2}}$ é o valor crítico. Neste projeto foi determinado um intervalo de confiança de 95% para a taxa de erro para cada classificador.

3.4. Teste de Friedman

No teste de Friedman os algoritmos são ranqueados de acordo com o seu desempenho (dos melhores para os piores) em cada conjunto de dados. A sua utilização exige a construção de duas hipóteses. A primeira (hipótese nula) é a de que o desempenho de todos os algoritmos é o mesmo. A segunda é a de que existe alguma diferença de desempenho no grupo de algoritmos testados. A primeira hipótese é rejeitada se a estatística calculada for maior que o valor crítico. Quando isso ocorre, um pós-teste é utilizado (neste projeto o Teste de Nemenyi).

3.5. Teste de Nemenyi

O teste de Nemenyi é um pós-teste, que nesse projeto foi usado como complemento do teste de Friedman, apenas quando a hipótese nula deste último fosse rejeitada. Esse teste tem o objetivo de encontrar quais algoritmos apresentaram diferença de desempenho.

3.6. Índice de Rand Corrigido

Mede a dissimilaridade entre uma partição a priori e uma partição obtida por um algoritmo de classificação.

V = {v1,...,vj,...,vC} - uma partição a priori
U = {u1,...,ui,...,uR} - uma partição obtida por um método

$$CR = \frac{\sum_{i=1}^{R}\sum_{j=1}^{C}\binom{n_{ij}}{2} - \binom{n}{2}^{2} - \binom{n}{2}^{-1}\sum_{i=1}^{R}\binom{n_{i.}}{2}\sum_{j=1}^{C}\binom{n_{.j}}{2}}{\frac{1}{2}\left[\sum_{i=1}^{R}\binom{n_{i.}}{2} + \sum_{i=1}^{R}\binom{n_{.j}}{2}\right] - \binom{n}{2}^{-1}\sum_{i=1}^{R}\binom{n_{i.}}{2}\sum_{j=1}^{C}\binom{n_{.j}}{2}}$$

Figura1: Equação para o índice de Rand Corrigido.

Onde:
nij representa o número de objetos que estão nas classes ui e vj;
ni. representa o número de objetos que estão na classe ui;
n.j representa o número de objetos que estão na classe vj;

4. Experimentos Realizados

Primeira Questão

Os dados da tabela Abalone foram aplicados ao algoritmo VKCM-K para obter uma partição com 3 clusters. Foram realizadas 100 execuções independentes do algoritmo, selecionando-se o melhor resultado segundo a função objetivo:

Melhor Execução = 53
Melhor J = 4088.354629856546

J(53) = 4797.949811802058
J(53) = 4175.465763602839
J(53) = 4100.434134142735
J(53) = 4089.921314272523
J(53) = 4088.652993309458
J(53) = 4088.516016870570
J(53) = 4088.466478833946
J(53) = 4088.431307096471
J(53) = 4088.394529329373
J(53) = 4088.390049367476
J(53) = 4088.362321386075
J(53) = 4088.354629856546

IRC(53) = 0.148535660402

Tabela 1: Tabela de contingência para a execução 53.

	v1	v2	v3	T
u1	674	625	229	1528
u2	651	516	140	1307
u3	47	426	869	1342
T	1372	1567	1238	4177

Tabela 2: Tabela de vetores de peso para a execução 53.

	1	2	3	4	5	6	7	8
Vetor 1	2.0818	2.0548	1.0364	0.8497	0.7325	0.7670	0.7553	0.6254
Vetor 2	1.4164	1.3733	0.7926	1.3641	1.0500	1.1696	1.0385	0.3728
Vetor 3	0.5197	0.5255	0.5663	1.9499	1.8003	1.9831	1.6620	0.5589

Segunda Questão

I. Máxima Verossimilhança

Supondo a função de densidade da estimação por máxima verossimilhança como sendo uma normal multivariada, e calculando-se as probabilidades a priori, determinou-se as probabilidades a posteriori de cada classe, afetando cada padrão à classe de maior probabilidade a posteriori.

Estimativa Pontual (Taxa de Erro) = 0.470194
IC(95%) = (0.455058, 0.485330)

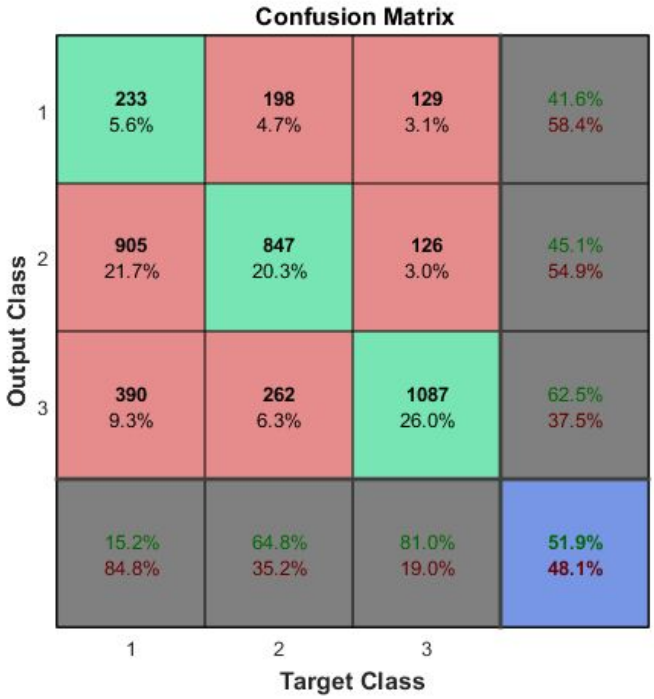

Figura 2: Matriz de confusão para o Classificador de Máxima Verossimilhança

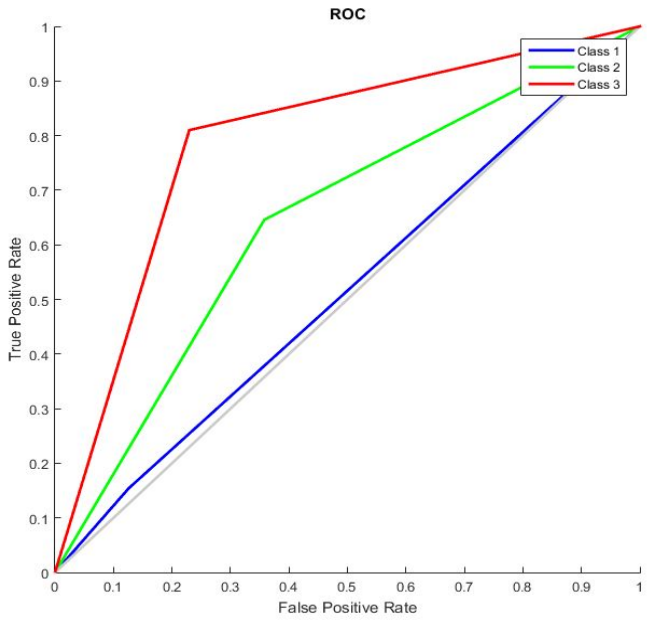

Figura 3: Curva ROC para o Classificador de Máxima Verossimilhança

II. MLP

Também foi utilizada a rede neural MLP, com apenas uma camada escondida para classificar os dados da base Abalone e três neurônios na camada de saída. Para encontrar uma arquitetura que tivesse uma menor taxa de erro, foi realizado uma variação na taxa de aprendizagem e na quantidade de neurônios na camada escondida com os seguintes valores [0.001 0.05 0.1 0.5 0.8] e [3 6 8 12 20], respectivamente. O treinamento da rede foi realizado através do algoritmo backpropagation por 1000 iterações. Através da técnica de validação cruzada foi escolhido 0.05 para a taxa de aprendizagem e 20 neurônios na camada escondida, pois apresentaram o menor erro de classificação dentre as possíveis combinações.

Melhor Taxa de Aprendizagem: 0.05000
Melhor Número de Neurônios: 20

Estimativa Pontual (Taxa de Erro) = 0.439071
IC(95%) = (0.424021, 0.454121)

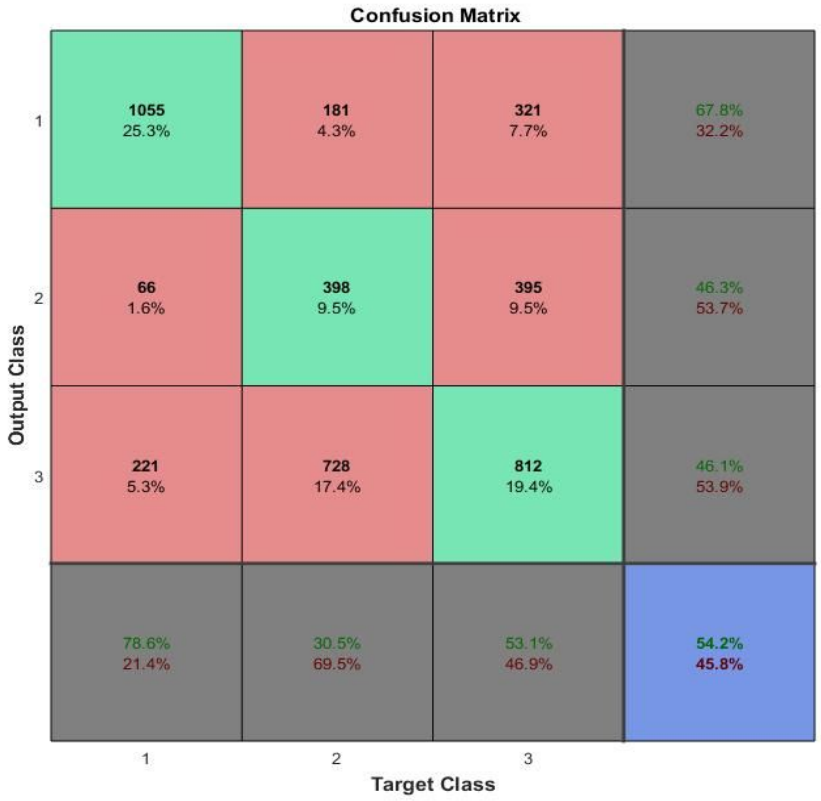

Figura 4: Matriz de Confusão para o MLP

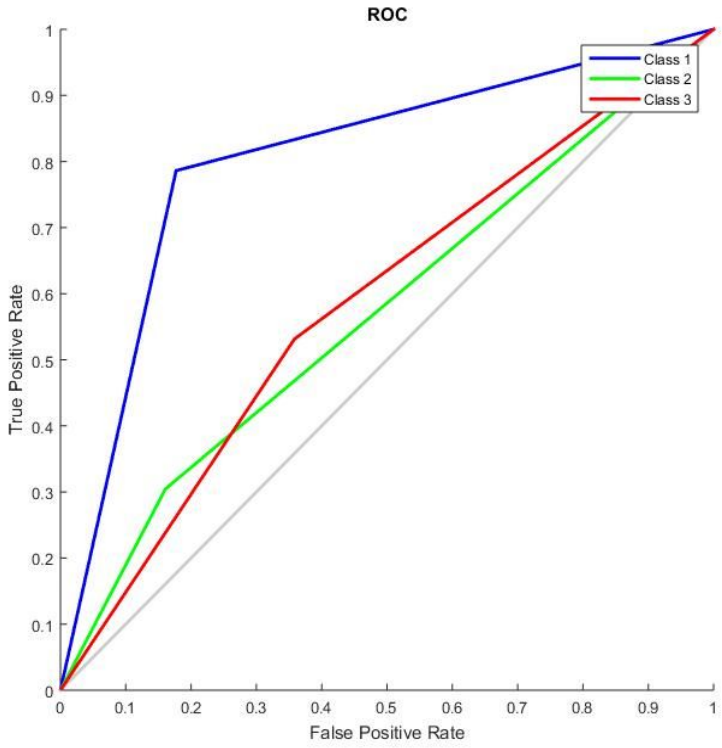

Figura 5: Curva ROC para o MLP

III. SVM

Na implementação do SVM através do Matlab, como função de *Kernel*, foi utilizada a função Gaussiana (*gaussian*), que obteve o melhor resultado dentre as funções apresentadas (*gaussian* ou *rbf, linear ou polinomial*). A função *fitcecoc* possibilita a classificação a partir de 3 rótulos diferentes (O SVM puro permite apenas a classificação binária). A função utilizada no projeto foi definida da seguinte maneira:

```
fitcecoc(X,Y,'Learners',t,'FitPosterior',1,'ClassNames',{'M','F','I'},'Verbose',2);
```

Na qual X e Y representam a matriz de atributos por exemplo, e vetor de rótulos, respectivamente; *t* representa o modelo SVM gerado através da função *templateSVM*; 'ClassNames' define que o próximo parâmetro passará o nome das classes (M,F,I). O valor do parâmetro C é ajustado internamente pela função, sendo 'tunado' para um valor ótimo que será utilizado na execução do algoritmo

Estimativa Pontual (Taxa de Erro) = 0.441705
IC(95%) = (0.426645, 0.456764)

IV. Voto Majoritário

Estimativa Pontual (Taxa de Erro) = 0.456069
IC(95%) = (0.440964, 0.471174)

Para cada classificador, foram obtidos uma estimativa pontual e um intervalo de confiança para a taxa de erro, conforme pode ser visto na tabela a seguir:

Tabela 7: Tabela de valores de estimativa pontual e intervalo de confiança por classificador

	Taxa de Erro	Intervalo de Confiança
MV	0.470194	(0.455058, 0.485330)
MLP	0.439071	(0.424021, 0.454121)
SVM	0.437635	(0.422590, 0.452680)
Voto Majoritário	0.441705	(0.426645, 0.456764)

O desempenho dos classificadores, foi calculado utilizando-se o teste de Friedman. Para isso foi necessário construir uma matriz de erros, que indica o erro de classificação de cada algoritmo em cada fold.

Tabela 8: Matriz de Erros

	MV	MLP	SVM	Voto Majoritário
Fold 1	0.5394	0.5322	0.4628	0.5346
Fold 2	0.5084	0.4630	0.4522	0.4558
Fold 3	0.4306	0.3971	0.4402	0.3995
Fold 4	0.4426	0.3947	0.4474	0.3852
Fold 5	0.4641	0.4474	0.4354	0.4450
Fold 6	0.4904	0.4713	0.4450	0.4378
Fold 7	0.4234	0.3732	0.4498	0.3804
Fold 8	0.4988	0.4676	0.4450	0.4772
Fold 9	0.4399	0.4159	0.4173	0.4255
Fold 10	0.4639	0.4279	0.4101	0.4351

A partir dessa matriz, foi calculado o p-value pelo teste de Friedman, e como ele (0.0087238) apresentou valor inferior a 0.05, a hipótese nula de que os algoritmos têm desempenhos significativamente iguais foi rejeitada, sendo necessário realizar o pós-teste de Nemenyi.

p-value: 0.0087238
α: 0.05
Como p-value(0.0087238) <= α(0.05), há diferenças significativas entre os algoritmos.

Teste de Nemenyi (α = 0.05)

No teste de Nemenyi, o desempenho de dois classificadores é diferente se a diferença entre seus ranks médios é maior ou igual ao valor crítico calculado a partir da fórmula:

$$CD = q_\alpha \sqrt{\frac{(k+1)}{6N}}$$

Onde q_α é o valor presente na tabela abaixo, para α = 0.05, k = 4 (número de modelos/algoritmos) e N = 10 (número de conjuntos de dados).

Critical values for the two-tailed Nemenyi test

# of models	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$q_{0.01}$	2.576	2.913	3.113	3.255	3.364	3.452	3.526	3.590	3.646	3.696	3.741	3.781	3.818	3.853
$q_{0.05}$	1.960	2.344	2.569	2.728	2.850	2.948	3.031	3.102	3.164	3.219	3.268	3.313	3.354	3.391
$q_{0.10}$	1.645	2.052	2.291	2.460	2.589	2.693	2.780	2.855	2.920	2.978	3.030	3.077	3.120	3.159

Tabela 9: valores críticos para o Teste de Nemenyi

Valor Crítico: 1.7973

Tabela 10: Matriz de Ranks

	MV	MLP	SVM	Voto Majoritário
Fold 1	4	2	1	3
Fold 2	4	3	1	2
Fold 3	3	1	4	2
Fold 4	3	2	4	1
Fold 5	4	3	1	2

Fold 6	4	3	2	1
Fold 7	3	1	4	2
Fold 8	4	2	1	3
Fold 9	4	1	2	3
Fold 10	4	2	1	3

Ranks Médios:

3.7000 2.0000 2.1000 2.2000

Tabela 11: Matriz de Diferença Absoluta de Ranks Médios

	MV	MLP	SVM	Voto Majoritário
MV	0	1.7000	1.6000	1.5000
MLP	1.7000	0	0.1000	0.2000
SVM	1.6000	0.1000	0	0.1000
Voto Majoritário	1.5000	0.2000	0.1000	0

Tabela 12: Diferença Absoluta >= Valor Crítico (α = 0.5)

	MV	MLP	SVM	Voto Majoritário
MV	0	0	0	0
MLP	0	0	0	0
SVM	0	0	0	0
Voto Majoritário	0	0	0	0

Com a tabela de Diferença Absoluta >= Valor Crítico (alpha = 0.05), é possível perceber que não é possível refutar que há um algoritmo melhor que outro.

5. Conclusão

O presente trabalho teve como objetivo consolidar os conceitos adquiridos na disciplina de Aprendizagem de Máquina, através da implementação de um algoritmo de agrupamento e de outros modelos de classificação utilizando a base de dados Abalone. Durante a implementação foram encontradas algumas dificuldades de implementação, principalmente por conta da complexidade da base de dados, devido a sua distribuição e dimensionalidade.

Para avaliação do desempenho dos algoritmos foram utilizadas algumas técnicas de avaliação e comparação dos modelos, como estimativa pontual, intervalo de confiança, validação cruzada e índice de rand corrigido.

Com os resultados obtidos, fora observado que os modelos apresentaram pequenas diferenças nas taxas de erro, destacando-se os modelos SVM e MLP por apresentarem os melhores resultados, apesar disso, através dos testes estatísticos foi possível perceber que os algoritmos apresentam desempenhos significativamente semelhantes.

6. Referências

[1] Carvalho, F. (2012). Aprendizagem de Máquina [http://www.cin.ufpe.br/~fatc/AM]. Universidade Federal de Pernambuco, Curso de Mestrado em Ciência da Computação.

[2] UC Irvine Machine Learning Repository (http://archive.ics.uci.edu/ml/)

[3] Duda, R. O., Hart, P. E., & Stork, D. G. (2001). Pattern classification (2nd ed). New York: Wiley–Interscience.

[4] Distribuição normal multivariada: (http://en.wikipedia.org/wiki/Multivariate_normal_distribution)

[5] Índice de Rand (http://en.wikipedia.org/wiki/Rand_index)

[6] Máxima Verossimilhança (http://en.wikipedia.org/wiki/Maximum_likelihood)

[7] SVM (https://en.wikipedia.org/wiki/Support_vector_machine)