**Relatório – Projeto Aprendizagem de Máquina**

**Aluno: Dion Maicon E. Duarte\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Escolaridade: ( ) Mestrado ( X ) Graduação**

**Observações:** Todas as informações necessárias para entendimento e execução dos algoritmos de Aprendizagem de Máquina implementados devem estar contidas nesse documento. Sejam criativos, concisos e claros utilizando figuras, fluxogramas, pseudocódigos ou outras ferramentas de visualização que acharem necessárias.

|  |
| --- |
| 1. **DESCRIÇÃO DAS BASES DE DADOS** |
| **1.1 Bases de Dados nº 1 – Base a ser usada para Clusterização** |
| **Nome da Base de Dados:**  labor.arff |
| **Local de Obtenção:**  uci |
| **Quantidade de Atributos:**  16 |
| **Quantidade de Amostras:**  57 |
| **Comentários:**  2 Classes de saída. São ignoradas na implementação do K-means. |
| **1.2 Bases de Dados nº 2 – Base a ser usada para Classificação** |
| **Nome da Base de Dados:**  vehicle.arff |
| **Local de Obtenção:**  uci datasets |
| **Quantidade de Atributos:**  18 |
| **Quantidade de Amostras:**  946 |
| **Comentários:**  4 Classes, não é possível utilizar a Perceptron |

|  |
| --- |
| **1.3 Bases de Dados nº 3 – Base a ser usada para Classificação** |
| **Nome da Base de Dados:**  labor.arff |
| **Local de Obtenção:**  **uci datasets** |
| **Quantidade de Atributos:**  16 |
| **Quantidade de Amostras:**  57 |
| **Comentários:**  Duas classes de saída |

|  |
| --- |
| 1. **PRÉ-PROCESSAMENTO**   <Informar os métodos utilizados para a normalização, imputação de valores e divisão da base de dados. Caso não tenha sido realizada alguma das etapas, preencher com “Não se aplica”.> |
| **2.1 Bases de Dados nº 1 – Base a ser usada para Clusterização** |
| **Normalização:**  Sim, Min-Max |
| **Imputação de valores:**  Sim, coloquei a média dos atributos de cada coluna em atributos faltantes |
| **Divisão da base de dados:**  57 amostras |
| * 1. **Bases de Dados nº 2 – Base a ser usada para Classificação** |
| **Normalização:**  Sim |
| **Imputação de valores:**  Não foi necessário, os atributos eram completos |
| **Divisão da base de dados:**  2/3 treinamento e 1/3 teste |
| * 1. **Bases de Dados nº 3 – Base a ser usada para Classificação** |
| **Normalização:**  Sim, Min-Max |
| **Imputação de valores:**  Sim, coloquei a media dos atributos nos atributos faltantes |
| **Divisão da base de dados:**  2/3 treinamento e 1/3 teste |

|  |
| --- |
| 1. **IMPLEMENTAÇÃO DO ALGORITMO** |
| **AGRUPAMENTO** |
| **k-means** |
| **Entrada:**  Informar o arquivo base .arff. E os k clusters para agrupamento |
| **Saída:**  k clusters contendo as n instancias devidamente calculadas pela distância euclidiana**.** |
| **Funcionamento do algoritmo:**  O algoritmo agrupa em k clusters as instâncias mais próximas do centroid. O centroid é escolhido inicialmente representado por k instãncias sorteadas aleatoriamente. Após as iterações são verificadas e ajustadas as posições dos centroids. O algoritmo para quando o número máximo de iterações é atingido ou os centroids permacerem por mais de uma iteração na mesma posição. |
| **Simulação do algoritmo:**  K: 3 Número Máximo de iterações: 10 Método: K-Means  ------------ Show Centroids --------  Centroid: 0, 0.0, 0.6, 0.4331428571428573, 0.6501240694789082, 0.52, 1.0, 0.625, 0.5416666666666666, 0.041666666666666664, 0.5, 0.3333333333333333, 0.0, 0.22222222222222232, 0.375, 0.0, 0.26,  Centroid: 1, 0.0, 0.74, 0.4331428571428573, 0.6501240694789082, 0.5, 1.0, 0.5, 0.5416666666666666, 0.125, 0.5, 0.3333333333333333, 1.0, 0.0, 0.0, 0.04761904761904767, 0.26,  Centroid: 2, 1.0, 0.3, 0.4, 1.0, 1.0, 0.7692307692307693, 0.625, 0.5416666666666666, 0.125, 0.5, 0.6666666666666666, 1.0, 0.22222222222222232, 0.0, 0.0, 0.0,  Clusters SIZE: 3  Numbers of Instances cluster 0: 24  Calc Centroid: 0, 0.0, 0.6, 0.4331428571428573, 0.6501240694789082, 0.52, 1.0, 0.625, 0.5416666666666666, 0.041666666666666664, 0.5, 0.3333333333333333, 0.0, 0.22222222222222232, 0.375, 0.0, 0.26,  Numbers of Instances cluster 1: 11  Calc Centroid: 1, 0.0, 0.74, 0.4331428571428573, 0.6501240694789082, 0.5, 1.0, 0.5, 0.5416666666666666, 0.125, 0.5, 0.3333333333333333, 1.0, 0.0, 0.0, 0.04761904761904767, 0.26,  Numbers of Instances cluster 2: 8  Calc Centroid: 2, 1.0, 0.3, 0.4, 1.0, 1.0, 0.7692307692307693, 0.625, 0.5416666666666666, 0.125, 0.5, 0.6666666666666666, 1.0, 0.22222222222222232, 0.0, 0.0, 0.0,  ------------ Show Centroids --------  Centroid: 0, 0.5833333333333334, 0.33416666666666667, 0.3880238095238095, 0.5935690653432589, 0.4875, 0.9030870445344131, 0.671875, 0.5277777777777777, 0.17621527777777782, 0.5416666666666666, 0.29950142450142453, 0.2713963963963964, 0.2870370370370371, 0.4583333333333333, 0.06547619047619051, 0.3266666666666666,  Centroid: 1, 0.4177489177489177, 0.6253246753246753, 0.5556883116883117, 0.6496729077374238, 0.5145454545454545, 0.8095325726904673, 0.5454545454545454, 0.537878787878788, 0.19917929292929293, 0.5454545454545454, 0.33333333333333337, 0.8206388206388208, 0.12121212121212127, 0.1590909090909091, 0.034632034632034674, 0.16363636363636364,  Centroid: 2, 0.875, 0.28, 0.4, 0.820409429280397, 0.625, 0.6277834008097166, 0.59375, 0.5885416666666666, 0.15863715277777776, 0.3125, 0.5496794871794871, 0.8175675675675678, 0.16666666666666674, 0.421875, 0.011904761904761918, 0.1925,  ----------------------------------------------------------------------  Clusters SIZE: 3  Numbers of Instances cluster 0: 22  Calc Centroid: 0, 0.5833333333333334, 0.33416666666666667, 0.3880238095238095, 0.5935690653432589, 0.4875, 0.9030870445344131, 0.671875, 0.5277777777777777, 0.17621527777777782, 0.5416666666666666, 0.29950142450142453, 0.2713963963963964, 0.2870370370370371, 0.4583333333333333, 0.06547619047619051, 0.3266666666666666,  Numbers of Instances cluster 1: 13  Calc Centroid: 1, 0.4177489177489177, 0.6253246753246753, 0.5556883116883117, 0.6496729077374238, 0.5145454545454545, 0.8095325726904673, 0.5454545454545454, 0.537878787878788, 0.19917929292929293, 0.5454545454545454, 0.33333333333333337, 0.8206388206388208, 0.12121212121212127, 0.1590909090909091, 0.034632034632034674, 0.16363636363636364,  Numbers of Instances cluster 2: 8  Calc Centroid: 2, 0.875, 0.28, 0.4, 0.820409429280397, 0.625, 0.6277834008097166, 0.59375, 0.5885416666666666, 0.15863715277777776, 0.3125, 0.5496794871794871, 0.8175675675675678, 0.16666666666666674, 0.421875, 0.011904761904761918, 0.1925,  ------------ Show Centroids --------  Centroid: 0, 0.5909090909090909, 0.30363636363636365, 0.3778441558441558, 0.5884277013309271, 0.48636363636363633, 0.9012697828487304, 0.7045454545454546, 0.5056818181818181, 0.17653093434343442, 0.5454545454545454, 0.2964257964257965, 0.27334152334152334, 0.3131313131313132, 0.48295454545454547, 0.06926406926406929, 0.3563636363636363,  Centroid: 1, 0.4304029304029304, 0.6321978021978022, 0.5471208791208791, 0.6497423172361138, 0.5123076923076924, 0.8270009342883835, 0.5096153846153846, 0.5737179487179488, 0.1951121794871795, 0.5384615384615384, 0.3333333333333333, 0.732848232848233, 0.10256410256410262, 0.16346153846153846, 0.032967032967033, 0.13846153846153847,  Centroid: 2, 0.875, 0.28, 0.4, 0.820409429280397, 0.625, 0.6277834008097166, 0.59375, 0.5885416666666666, 0.15863715277777776, 0.3125, 0.5496794871794871, 0.8175675675675678, 0.16666666666666674, 0.421875, 0.011904761904761918, 0.1925,  ----------------------------------------------------------------------  Centroid movimentou: false, Numero de passos: 2  Algoritmo Executou com Sucesso!! |
| **Medida de desempenho utilizada:**  Não utilizei uma avaliação dos k clusters. |
| **CLASSIFICAÇÃO** |
| **kNN** |
| **Entrada:**  Dois arquivos um para treinamento e um para teste e o valor de k |
| **Saída:**  n amostras de teste avaliadas e a taxa de acerto |
| **Funcionamento do algoritmo:**  n entradas para treinamento e m entradas para teste, para cada elemento do conjunto de teste calcular a distancia para cada elemento do conjunto de treinamento, classificar pela menor distância e apresentar a classe predita x esperada. |
| **Simulação do algoritmo:** |
| **Medida de desempenho utilizada:**  Matriz de Confusão com taxa de acerto**.** |
| **Árvore de decisão** |
| **Entrada:** |
| **Saída:** |
| **Funcionamento do algoritmo:** |
| **Simulação do algoritmo:** |
| **Medida de desempenho utilizada:** |
| **Redes Neurais** |
| **Entrada:**  Dois arquivos um para treinamento e outro para teste. Os parametros ficaram fixados não deixei uma interface para alterar erro e número de iterações. |
| **Saída:**  n amostras de teste avaliadas e a taxa de acerto. |
| **Funcionamento do algoritmo:**  São atribuidos pesos aleatorios para cada atributo, os atributos são então somados e passam por uma função de ativação, se a ativação não ocorreu como o esperado, os pesos são recalculados utilizando uma taxa de erro os atributos atuais e o peso dos atributos. O algoritmo tem 2 críterios de parada: máximo de iterações e erro máximo. |
| **Simulação do algoritmo:**  Método: RN  Instancias Treinadas: -------------------------  0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0  0.0 0.0 1.0 0.0 0.0 0.0 0.0 1.0 1.0 0.0  0.0 1.0 0.0 0.0 0.0 1.0 0.0 0.0 0.0 1.0  0.0 1.0 1.0 1.0 1.0 0.0 0.0 0.0 0.0 0.0  0.0 0.0 0.0  Printar a curva de erro para as iterações:  Iteração: 1, erro: 0.46511627906976744  Iteração: 2, erro: 0.18604651162790697  Iteração: 3, erro: 0.16279069767441862  Iteração: 4, erro: 0.09302325581395349  Iteração: 5, erro: 0.06976744186046512  Iteração: 6, erro: 0.09302325581395349  Iteração: 7, erro: 0.06976744186046512  Iteração: 8, erro: 0.09302325581395349  Iteração: 9, erro: 0.11627906976744186  Iteração: 10, erro: 0.09302325581395349  Iteração: 11, erro: 0.046511627906976744  Iteração: 12, erro: 0.046511627906976744  Iteração: 13, erro: 0.09302325581395349  Iteração: 14, erro: 0.023255813953488372  Iteração: 15, erro: 0.046511627906976744  Iteração: 16, erro: 0.046511627906976744  Iteração: 17, erro: 0.0  Pesos gerados: [-0.14928332722681809 0.17222937436829205 -0.9927199724424207 -0.7900980578024196 0.21390158532983206 0.07595525152907028 0.49400203653682895 -0.2944505856503754 -0.06380634035102192 0.13607203116665575 -0.4025659622435179 -0.32328337969175863 0.2893129768355086 1.012947901399378 -0.013016257356637084 0.42653570485103987 0.06376762434861355 ]  Instancias do Teste: ----------------------------  Classificado como: 1.0, esperado: 1.0  Classificado como: 1.0, esperado: 1.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 1.0, esperado: 0.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 0.0, esperado: 1.0  Classificado como: 1.0, esperado: 1.0  Classificado como: 1.0, esperado: 1.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Classificado como: 0.0, esperado: 0.0  Verdadeiros positivos: 12.0, total de instãncias de teste: 19.0, taxa de acerto: 63.15% |
| **Medida de desempenho utilizada:**  O algoritmo funciona até o criterio de parada, na maioria dos testes a quantidade de iterações nas bases testadas não passou de 30. O Perceptron simples aceita somente duas classes. Testei com quatro classes e tentei adaptar, mas não obtive uma quantidade de acerto aceitável. As classes são convertidas em atributos númericos para uso do algoritmo. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. **RESULTADOS OBTIDOS** | | | | |
| **AGRUPAMENTO** | | | | |
| **k-means** | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | **K = 1** | **K = 3** | **K = 5** | **K = 7** | | | | | |
| |  |  |  |  | | --- | --- | --- | --- | | As 43 instâncias do conjunto formaram um grupo. | Executou 3 iterações, com k grupos de 28, 10 e 5 classes. | Executou 6 iterações com k grupos de 14, 4, 1, 17 e 7 classes. | Executou 6 iterações com k grupos de 11, 5, 4, 7, 5 e 5 classes. | | | | | |
| **CLASSIFICAÇÃO** | | | | |
| **kNN** | | | | |
| **Base de dados** | **K = 1** | **K = 3** | **K = 5** | **K = 7** |
| **1** | Verdadeiros positivos: 18.0, total de instâncias de teste: 19.0, taxa de acerto: 94.73% | Verdadeiros positivos: 18.0, total de instâncias de teste: 19.0, taxa de acerto: 94.73% | Verdadeiros positivos: 16.0, total de instâncias de teste: 19.0, taxa de acerto: 84.21% | Verdadeiros positivos: 16.0, total de instâncias de teste: 19.0, taxa de acerto: 84.21% |
| **2** | Verdadeiros positivos: 146.0, total de instancias de teste: 211.0, taxa de acerto: 69.19% | Verdadeiros positivos: 146.0, total de instancias de teste: 211.0, taxa de acerto: 69.19% | Verdadeiros positivos: 149.0, total de instancias de teste: 211.0, taxa de acerto: 70.61% | Verdadeiros positivos: 150.0, total de instancias de teste: 211.0, taxa de acerto: 71.09% |
| **Árvore de decisão** | | | | |
| **Base de dados 1** | | | | |
|  | | | | |
| **Base de dados 2** | | | | |
|  | | | | |
| **Redes neurais** | | | | |
| **Base de dados 1** | | | | |
| Verdadeiros positivos: 14.0, total de instâncias de teste: 19.0, taxa de acerto: 73.68%. O conjunto está bem dividido e foi realizado a imputação de valores a taxa é aceitável devido ao conjunto e quantidade de atributos. Os melhores resultados variam entre 60% e 75%. | | | | |
| **Base de dados 2** | | | | |
| Verdadeiros positivos: 62.0, total de instâncias de teste: 211.0, taxa de acerto: 29.38%. A rede perceptron simples aceita somente duas classes,mas está tem 4 classes. | | | | |

|  |
| --- |
| 1. **CONCLUSÃO** |
| **5.1 Comparativo de desempenho entre as bases de dados** |
| As bases de dados escolhidas foram bem diferentes, a base 1 continha muitos dados faltantes e mesclava atributos númericos com categóricos. A base 2 continha os dados completos, mas uma quantidade de classes grande o que impossibilita o uso da Perceptron, mas ajudou muito a entender os conceitos de entropia e as justificativas de uso de camadas ocultas em outros tipos de RNA. |
| **5.2 Comparativo de desempenho entre os algoritmos implementados** |
| O único algoritmo de agrupamento ajudou a entender os conceitos, as bases utilizadas continham classes, logo, era possivel saber de antemão quais eram os grupos corretos, quando utilizei a base 2 no K-Means com k = 4, foi possível perceber que o algoritmo comportou-se de maneira adequada pois tinha uma boa distribuição das classes na proporção da base de treinamento.  Ao utilizar a base 1 no k-Means com o valor de k = 2 o resultado também foi satisfatório. Conhecer os dados é a chave para saber qual o melhor algoritmo.  O K-NN é um algoritmo simples mas com um custo alto, apresentou um bom resultado para a base 1 com duas classes e um resultado não tão bom com a base 2.  A rede neural correspondeu bem ao conjunto de dados 1, a taxa de acerto reflete muito a falta de atributos da base, acredito que se a base contem um número menor de imputações o resultado seria melhor. A base de dados 2 correspondeu como o esperado e apresentou um resultado esperado já que continha 4 classes e a rede perceptron trabalha com apenas 2 classes. |