PUCRS – FACIN

**Inteligência Artificial – Engenharia da computação**

**Alunos: Guilherme Korol e Matheus Storck**

Prof.: Silvia Moraes

**T4 – Aprendizagem de Máquina**

1. **Problema e objetivos**

O problema proposto, conforme enunciado, é o reconhecimento de dígitos manuscritos. Para tal, iremos explorar quatro técnicas de aprendizado de máquina. O *dataset* disponibilizado foi dividido em conjunto de treinamento e teste. Adicionalmente, o conjunto de teste foi incrementado com amostras editadas manualmente afim de sofisticar a etapa de generalização. Com este trabalho, buscamos o aprofundamento prático no tópico de aprendizagem de máquina, complementando a fundamentação teórica vista durante as aulas deste assunto.

1. **Conjuntos de dados e Pré-processamento**

Originalmente, o conjunto de dados foi gerado a partir da coleta de dígitos manuscritos de 43 pessoas. Onde 30 delas contribuíram para o conjunto de treino e as restantes para o conjunto de teste. Cada amostra foi então pré-processada (por um filtro passa-baixas e *subsampling*) pelos autores de forma a extrair bitmaps normalizados em formato 32x32.

Para reduzir dimensionalidade, as amostras de 32x32 foram divididas em 64 janelas, sem sobreposição, de tamanho 4x4. Essa divisão gera amostras de tamanho 8x8, onde cada atributo é o número total de valores um na respectiva janela 8x8 da matriz de entrada. Já para adaptação para o formato Weka, cada amostra de 8x8 atributos, mais a classe, foram formatadas de acordo. Formando, então, os arquivos de entrada para treinamento e teste.

Portanto, o conjunto total foi dividido em conjunto de treinamento e conjunto de teste. O primeiro, é constituído de 3823 amostras (68% do conjunto), enquanto o segundo possuí 1797 amostras (32%).

1. **Algoritmos e parâmetros de treinamento**

Como a tarefa proposta consiste em um problema de classificação, escolhemos os algoritmos Rede Multi-Layer Perceptron, Rede Bayesiana, Árvore de Decisão e Regressão Logística Multinominal para avaliação neste trabalho.

* 1. **Rede Multi-Layer Perceptron**

A rede Multi-Layer Perceptron (MLP) é uma rede neural artificial do tipo *feedforward*, composta de três ou mais camadas (de entrada, saída e demais camadas ocultas).

As camadas ocultas são compostas de neurônios artificiais totalmente conectados as camadas vizinhas. Cada neurônio se resume a uma soma ponderada de todas suas entradas por determinados pesos, além de um termo bias. O resultado, então, é atribuído a uma função de ativação. Existem diversos tipos de funções de ativação, as mais usadas são sigmoide e tangente hiperbólica.

A aprendizagem em uma rede Multi-Layer Perceptron consiste em refinar os pesos de cada neurônio para que a camada de saída apresente a classe correta. O processo é realizado pelo algoritmo *Error Backpropagation*.

* 1. **Redes Bayesianas**

As redes Bayesianas utilizam probabilidade e estatística de modo a mapear o relacionamento entre os atributos de entrada em um grafo acíclico dirigido. Cada nodo da rede representa um evento, e as ligações entre os nodos são as probabilidades que relacionam os eventos nodo.

No caso mais simples, a rede pode ser construída por um especialista na área de aplicação e parte-se diretamente para generalização. Por outro lado, quando o método mais simples não é possível, deve-se usar algum algoritmo de aprendizado de máquina para *aprender* as relações entre atributos. Os algoritmos usados diferem quanto a forma e o que objeto da aprendizagem. Em redes bayesianas, ora se tem a topologia do grafo e é preciso aprender os parâmetros, ora é preciso aprender topologia e parâmetros.

Iremos avaliar o desempenho de redes bayesianas com diferentes algoritmos de busca para encontrar a topologia. O algoritmo que encontra a topologia em redes bayesianas é o seguinte:

1. Lê entrada (*dataset*, Métricas de avaliação ou *Scores* e possíveis topologias).

2. Inicializa a estrutura (vazio, randomicamente, árvore, etc.)

3. Repete até que *score*(;D) = *score*(;D)

4. Altere um vértice que resulte na melhor melhora de *score* possível entre todas operações possíveis.

5. =

Onde G é um grafo (rede bayesiana) e D o *dataset* de entrada. Os algoritmos explorados são aplicados no passo 4.

* 1. **Árvores de Decisão**

Basicamente, o algoritmo parte de uma modelagem de conjunto de decisões específicos (e suas consequências) em uma árvore. Árvores de decisão podem apresentar três tipos de nodos: nodos de decisão, de chance, e nodos finais (folhas).

Para nosso exemplo de classificação, a árvore deve conter dez folhas (uma para cada classe). Um dos algoritmos mais populares para aprendizagem em árvores de decisão é o de Partição Recursiva (*Recursive Partioning*). Em linhas gerais, o algoritmo recursivamente reparte o conjunto de dados isolando variáveis independentes até que o conjunto retorne a mesma classe para todos indivíduos. Dessa forma, o algoritmo aprende os “critérios de decisão” que devem estar nos vértices para classificar corretamente.

* 1. **Regressão Logística Multinominal**

Este algoritmo é uma generalização da versão original da regressão logística. Os modelos de regressão logística procuram encontrar uma função que separa linearmente o plano em duas regiões ou classes. A regressão multinominal parte do modelo linear, acrescentando variáveis independentes e coeficientes de regressão que tornem possível a classificação em mais de duas classes.

Da mesma forma que em muitas das técnicas de classificação estatística, o objetivo da aprendizagem é encontrar o conjunto de pesos (coeficientes) que produza o menor erro. Os algoritmos de aprendizagem são herdados da regressão logística com a diferença de que as variáveis de classificação agora são categóricas e não mais binárias. Alguns exemplos de algoritmos são *Maximum a posteriori estimation* (MAP) e *Maximum Likelihood estimation* (MLE).

1. **Análise dos resultados**
   1. **Rede Multi-Layer Perceptron**

Para rede MLP, treinamos com diversos número de épocas e para dois valores de taxa de aprendizagem. O conjunto de teste foi utilizado para avaliar o erro em todos os resultados apresentados nesse trabalho. A Figura 1 apresenta as taxas da Raiz do Erro Médio Quadrático (RMSE em inglês). Usamos RMSE por se tratar de uma métrica que informa “quão concentrado” as predições estão entorno da classe correta (acurácia).

1. Figura 1 - RMSE para taxas de aprendizagem de 0.3 e 0.5.

Podemos observar que os melhores resultados foram obtidos com taxa de 0.5 e que a partir de, aproximadamente, 1000 épocas o erro estabiliza para a mesma taxa de aprendizagem. Porém, com apenas 100 épocas atinge o menor valor. Treinando para número de épocas maiores que 100 observamos *overfitting* do modelo. Por esse motivo, iremos apresentar na Tabela 1 precisão, abrangência e F1 para o modelo aprendido com 100 épocas e taxa de aprendizagem de 0.5.

Além do mais, acreditamos que o erro para taxa de aprendizagem de 0.3 poderia diminuir com treinamentos mais longos (mais épocas). Objetivamente, também estamos observando na Figura 1 a diferença na velocidade de aprendizado com que as duas taxas convergem.

|  |  |  |  |
| --- | --- | --- | --- |
| **Classe** | **Precisão** | **Abrangência** | **F1** |
| 0 | 0.994 | 0.989 | 0.992 |
| 1 | 0.948 | 0.995 | 0.971 |
| 2 | 0.978 | 0.994 | 0.986 |
| 3 | 1 | 0.951 | 0.975 |
| 4 | 0.973 | 0.983 | 0.978 |
| 5 | 0.937 | 0.984 | 0.96 |
| 6 | 1 | 0.989 | 0.994 |
| 7 | 0.982 | 0.922 | 0.951 |
| 8 | 0.975 | 0.908 | 0.94 |
| 9 | 0.917 | 0.978 | 0.946 |
| Média: | 0.97 | 0.969 | 0.969 |

Tabela 1 - Valores de Precisão, Abrangência e F1 por classe. 100 Épocas e 0.5 de taxa de aprendizagem.

* 1. **Redes Bayesianas**

Na classificação com redes bayesianas, exploramos diferentes algoritmos de busca para encontrar a topologia da rede. Iremos comparar os algoritmos K2, *Hill* Climber, Repeated *Hill Climber* e *LAGD Hill Climber*. Outros algoritmos que poderiam ser usados, como *Simulated Annealing* e Algoritmos Genéticos, não executaram nas máquinas testadas.

Os algoritmos testados não apresentaram diferenças nos resultados da classificação com redes bayesianas. Na Tabela 2, está representado precisão, abrangência e índice F1 para todas classes na classificação por rede bayesiana.

|  |  |  |  |
| --- | --- | --- | --- |
| **Classe** | **Precisão** | **Abrangência** | **F1** |
| 0 | 0.989 | 0.978 | 0.983 |
| 1 | 0.833 | 0.824 | 0.829 |
| 2 | 0.895 | 0.87 | 0.883 |
| 3 | 0.94 | 0.858 | 0.897 |
| 4 | 0.887 | 0.95 | 0.917 |
| 5 | 0.961 | 0.94 | 0.95 |
| 6 | 0.989 | 0.956 | 0.972 |
| 7 | 0.922 | 0.927 | 0.925 |
| 8 | 0.843 | 0.833 | 0.838 |
| 9 | 0.783 | 0.883 | 0.83 |
| Média: | 0.904 | 0.902 | 0.903 |

Tabela 2 - Valores de Precisão, Abrangência e F1 por classe para rede bayesiana.

* 1. **Árvores de Decisão**
  2. **Regressão Logística Multinominal**

1. **Conclusões**

|  |  |  |  |
| --- | --- | --- | --- |
| **Técnica** | **Precisão Média** | **Abrangência Média** | **F1**  **Médio** |
| MLP | 0.97 | 0.969 | 0.969 |
| Rede Bayesiana | 0.904 | 0.902 | 0.903 |
| Árvore de Decisão | - | - | - |
| K-NN | - | - | - |

1. **Referências**

E. Alpaydin, C. Kaynak. Optical Recognition of Handwritten Digits Data Set. Disponível em: <https://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits> Acesso em: 29 de Junho de 2018