Aprendizagem de Máquina (2020/Período Especial) - Impactos da Base de Aprendizagem

Diogo C. T. Batista[1]

[1]Universidade Federal do Paraná (UFPR)
Curitiba – Paraná – Brasil

diogo@diogocezar.com

1. Impactos da Base de Aprendizagem

Esta atividade laboratorial tem como objetivo a investigação exploratória dos impáctos da base de aprendizagem na performance de diferentes classificadores. Os classificadores a serem analisados neste laboratório são:

- KNN;
- Naïve Bayes;
- Linear Discriminant Analysis;
- Logistic Regression;
- Perceptron.

Os dados estão dispostos em arquivo no formato *svmlight*. Este formato dispõe as informações de categorias e suas respectivas características. O código 1 mostra um pequeno trecho desta representação.

```
1   0 1:0.000000 2:0.000000 3:0.001412 4:0.000000 5:0.014124 6:0.000000 ...
```

Código 1. Exemplo do Formato de Entrada

Neste exemplo, pode-se notar que o 0 inicial mostra a qual categoria este dado pertence, na sequência, demonstra-se para cada característica o seu valor. A característica 1 possui o valor 0.000000.

Os dados utilizados neste experimento estão divididos em 2 partições. A primeira partição possui 20.000 registros, que foi definida para uma base de treinamento. Já a segunda, possui 58.646, utilizados para o teste dos classificadores.

1.1. Estratégia para execução do experimento

Para automatização dos experimentos, foi criado um sistema *orquestrador* no qual é possível definir as instruções para a execução de todo o fluxo para os 5 diferentes classificadores. O Código 2 mostra os valores definidos em um arquivo no formato *JSON*.

```
[
  {
    "classifier": "knn",
    "chunk_start": 1000,
    "chunk_stop": 20000,
    "chunk_step": 1000
  },
  {
    "classifier": "naive\_bayes",
    "chunk_start": 1000,
    "chunk_stop": 20000,
    "chunk_step": 1000
  },
  ...
]
```

Código 2. JSON do Orquestrador

Neste JSON, *classifier* é o nome do classificador que deve ser utilizado. Estes, podem variar entre: *knn*, *naive_bayes*, *lda*, *logistic_regression*, e *perceptron*. Os outros atributos servem parar criar partes (*chunks*) que dividem os testes em função da disponibilidade da base de treinamento. Por exemplo, no primeiro bloco, *chunk_start* indica o tamanho do bloco inicial, *chunk_stop* indica a condição de parada, e por fim o *chunk_step* indica o incremento a cada passo.

Para os testes, em todos os classificados variou-se a base de treinamento de 1000 até 20000, com o intervalo de 1000.

O *script* implementado ainda realiza a automação da coleta dos resultados, salvando arquivos no formato *csv* todos os dados extraídos dos experimentos. Uma tabela armazena os resultados de todos os experimentos, tabulando: *Classifier*, *Chunk*, *F1Score*, *Accuracy* e *Execution Time (s)*. Além disso, para cada uma das execuções é criado um arquivo separado com a sua matriz de confusão.

1.2. Parametrização dos Classificadores

Para este trabalho, foram implementados os parâmetros padrões para todos os classificadores anteriormente descritos. Assim sendo, nenhum ajuste específico foi realizado qualquer um dos classificadores testados.

1.3. Experimentos

Foram realizados 100 experimentos. 20 variações (entre 1000 20000) para cada um dos 5 classificadores. Os 20 melhores resultados estão demonstrados por ordem de *F1Score* seguidos de *Acurácia* e o *Tempo de Execução* na Tabela 1.

Classifier	Chunk	F1Score	Accuracy	Execution Time (s)
knn	20.000	0,939	0,939	292,836
perceptron	11.000	0,938	0,938	5,501
knn	19.000	0,938	0,938	279,718
knn	18.000	0,938	0,937	268,223
knn	16.000	0,937	0,937	251,132
knn	17.000	0,937	0,937	257,359
knn	12.000	0,936	0,936	225,239
knn	13.000	0,936	0,936	237,264
knn	14.000	0,936	0,936	253,580
knn	15.000	0,936	0,935	270,468
perceptron	17.000	0,935	0,935	5,710
knn	10.000	0,935	0,935	198,619
knn	11.000	0,935	0,934	210,219
perceptron	18.000	0,934	0,934	5,717
knn	9.000	0,933	0,933	178,040
knn	8.000	0,930	0,929	157,567
lda	10.000	0,928	0,928	5,030
perceptron	13.000	0,928	0,928	5,337
lda	20.000	0,928	0,928	5,403
perceptron	20.000	0,926	0,927	5,677

Tabela 1. Melhores Resultados

1.4. Análises dos Resultados

A análise dos resultados leva em considerações os quesitos:

1. Comparação do desempenho dos classificadores;
2. Classificador que tem o melhor desempenho com poucos dados;
3. Classificador que tem melhor desempenho com todos os dados;
4. Classiticador mais rápido;
5. Análise das matrizes de confusão;

1.4.1. Comparação do desempenho dos classificadores

O gráfico da Figura 1 dispõe no eixo X a variação dos *chunks* testados. O eixo Y é formado pela variação dos resultados obtidos por F1Score. E as linhas representam os 5 diferentes classificadores.

Destaca-se o comportamento do classificador *perceptron*, que apesar da tendência crescente, mostra uma variação de resultados durante todos os experimentos. Isso acontece por sua por sua característica natual chamada de *Cathastrophic Forgetting*, que acaba "esquecendo" os resultados anteriores quando não são frequentemente revisitados nos cálculos. Nota-se ainda, que ao relizar treinamentos com *chunks* maiores, entre 15.000 e 20.000, as variações são menores e os resultados são melhores.

Em geral para os classificadores *logistic_regression*, *naive_bayes*, *lda* e *knn* a estabilização na melhora dos resultaods aconteceu nos testes com 10.000 registros usados para o treinamento.

1.4.2. Classificador que tem o melhor desempenho com poucos dados

Ainda analisando o gráfico da Figura 1, pode-se notar que o classificador que obteve melhor desempenho com poucos dados foi o *perceptron*, seguido do *lda* e *knn*.

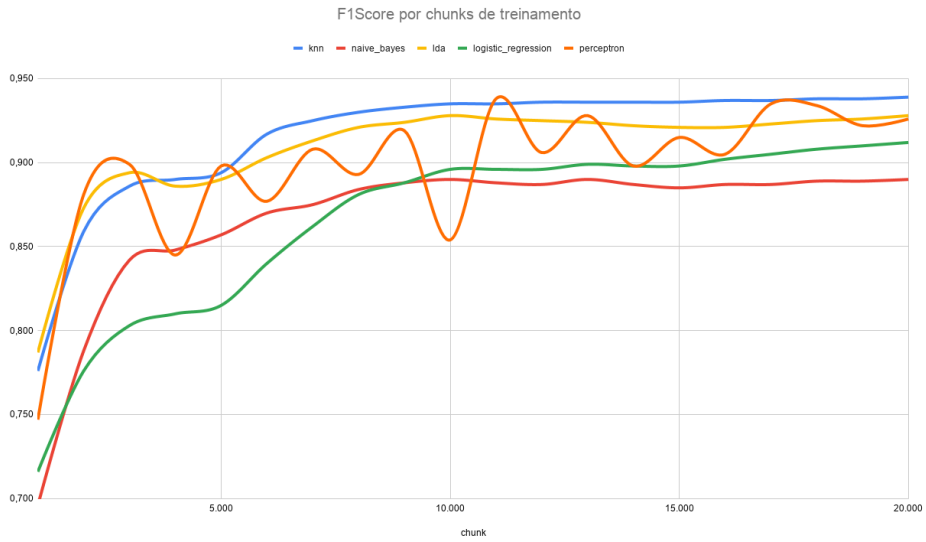

Figura 1. Comparação dos Classificadores

1.4.3. Classificador que tem melhor desempenho com todos os dados

O classificador *knn* conseguiu o melhor resultado com todos os dados, conseguindo um F1Score e Acurácia de $0,939$, utilizando os 20.000 dados de treinamento.

1.4.4. Classiticador mais rápido

O classificador mais rápido foi o *perceptron*, que com em apenas $5,5s$ conseguiu obter F1Score e Acurácia de $0,938$.

1.5. Análise das matrizes de confusão

Para a análise das matrizes de confusão, foram consideradas as matrizes que tiveram o treinamento com toda a base (20.000 registros). Excluindo a diagonal com os acertos, criou-se os gráficos da Figura 2, no qual a variação de cor das células (em uma escala de azul) é definida por seu valor.

As imagens analisadas são representações manuscritas de dígitos de 10 categorias diferentes, que representam os número decimais $0, 1, 2, 3, 4, 5, 6, 7, 8, 9$. Com isso, observa-se que em várias células os erros são compartilhados entre os classificadores. Isso se deve pois, as confusões são encontradas em números com o formato semelhante, como por exemplo, a confusão entre $(3, 5)$ que possui um alto índice de falhas nos classificadores *knn*, *lda* e *logistic_regression*, mas menos expressivo nos classificadores *naive_bayes* e *perceptron*.

Como destaques pode-se observar que o *knn* é o que apresenta as falhas de forma distribuída.

Outro destaque é o classificador *perceptron* que possui os erros distruídos de forma esparçada, com muitos erros do mesmo tipo.

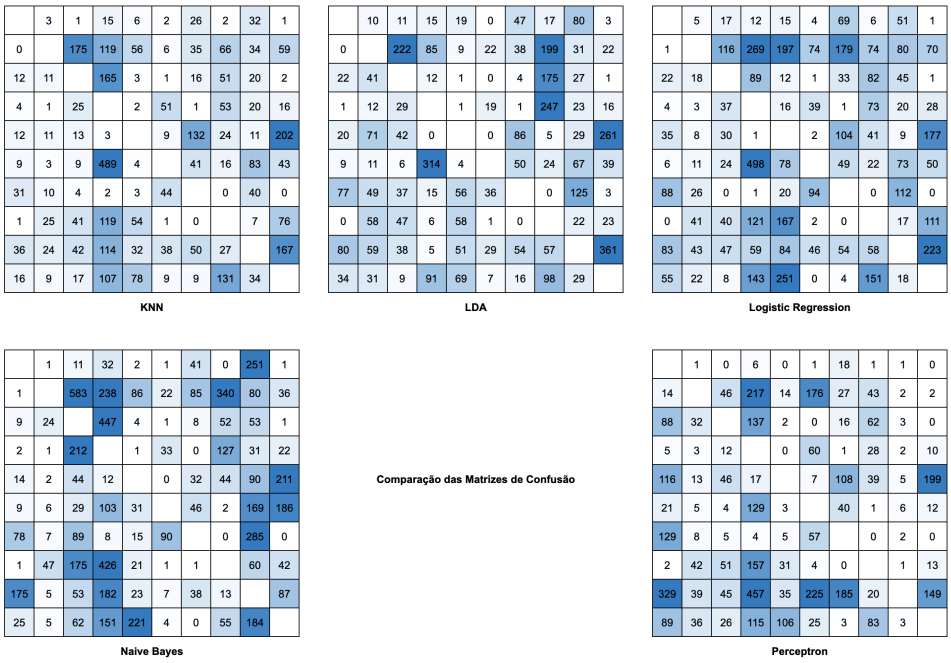

Figura 2. Comparação das Matrizes de Confusão

1.6. Código Fonte

Os códigos preparados podem ser analisados através do repositório: https://github.com/diogocezar/machine-learning/tree/master/lab2/src