Aprendizagem de Máquina - Laboratório 2

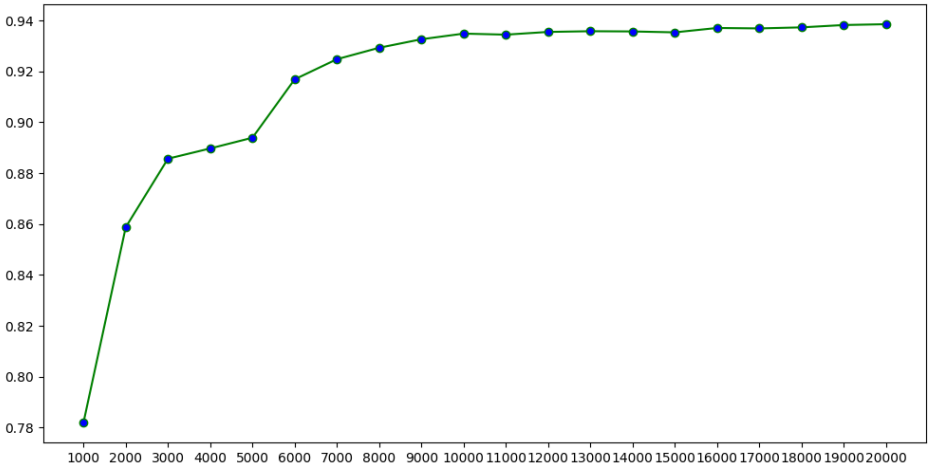
Maria Teresa Kravetz Andrioli

Universidade Federal do Paraná (UFPR)
Curitiba – PR – Brasil

Repositório github com programa usado e dados extraídos:
https://github.com/mariaandrioli/aprendizagem-de-maquina/tree/main/L2

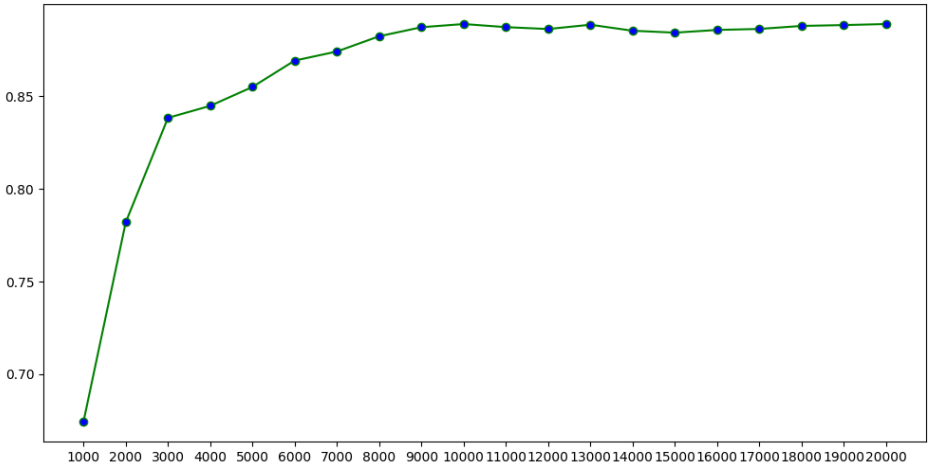
1. Comparação em função da base de treinamento

1.1. KNN



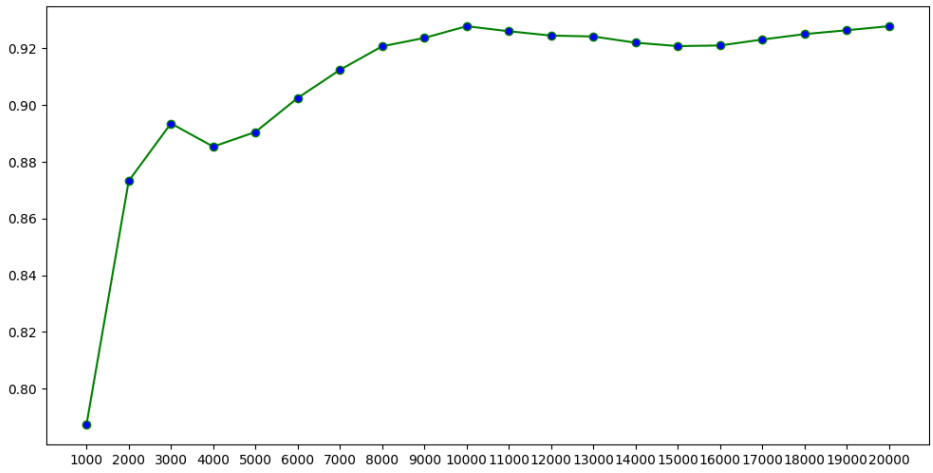
A curva do gráfico tem tendências de ficar mais reta a partir de 9000 dados.

1.2. Naive Bayes



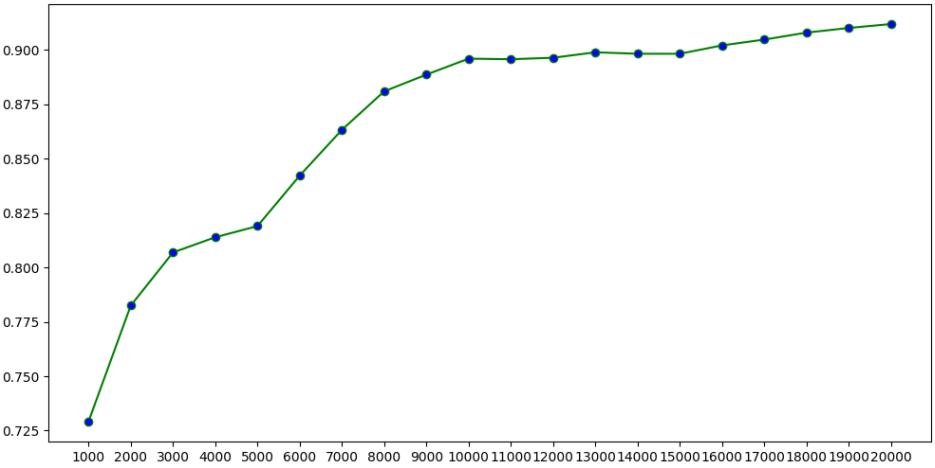
A curva do gráfico tem tendências de ficar mais reta a partir de 9000 dados.

1.3. LDA



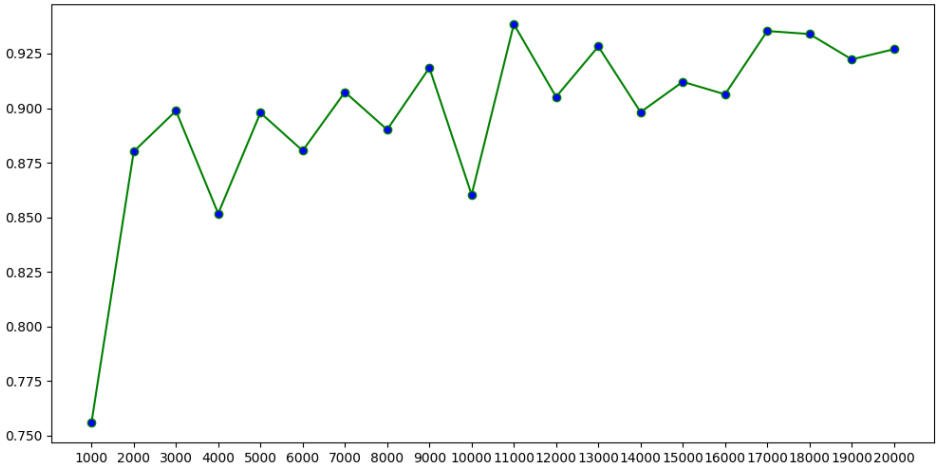
A curva do gráfico tem tendências de ficar mais reta a partir de 10000 dados, mas tem um curva significante para baixo no marco de 15000 dados.

1.4. Logistic Regression



A curva do gráfico tem tendências de ficar mais reta a partir de 10000 dados, mas continua até 20000 com um crescimento constante.

1.5. Perceptron



A curva do gráfico não tem tendência de ficar mais reta, com bastante variação.

2. Melhor desempenho com poucos dados (1000 dados)

Classificador	Precisão (arredondado para 3 casas)	Tempo (sys+user)
KNN	0.782	66.76s
Naïve Bayes	0.674	13.95s
Linear Discriminant Analysis	0.787	11.01s
Logistic Regression	0.729	16.39s
Perceptron	0.756	12.39s

Com o parâmetro de 1000 linhas na base de dados de treinamento, o classificador com melhor desempenho foi o Linear Discriminant Analysis, com o menor tempo e maior índice de precisão.

3. Melhor desempenho com todos os dados

Classificador	Precisão (arredondado para 3 casas)	Tempo (sys+user)
KNN	0.939	233.55s
Naïve Bayes	0.889	15.02s
Linear Discriminant Analysis	0.928	13.69s

Logistic Regression	0.912	43.03s
Perceptron	0.927	10.74s

Com todos os dados, a comparação fica um pouco mais complicada, pois o KNN tem uma precisão levemente mais alta, porém um tempo muito maior (17 vezes maior). Arredondando mais uma casa, a precisão ficaria 0,94 para o KNN e 0,93 para o LDA. por causa disso, o desempenho do LDA é melhor. No entanto, acredito que se a diferença de precisão fosse um pouco maior e o tempo de resolução do problema não fosse tão relevante, existem casos que o KNN possa ser o escolhido.

4. Análise matrizes de confusão

4.1. KNN

```
[[5472    3    1   15    6    2   26    2   32    1]
 [   0 6105  175  119   56    6   35   66   34   59]
 [  12   11 5607  165    3    1   16   51   20    2]
 [   4    1   25 5646    2   51    1   53   20   16]
 [  12   11   13    3 5305    9  132   24   11  202]
 [   9    3    9  489    4 4842   41   16   83   43]
 [  31   10    4    2    3   44 5724    0   40    0]
 [   1   25   41  119   54    1    0 5773    7   76]
 [  36   24   42  114   32   38   50   27 5165  167]
 [  16    9   17  107   78    9    9  131   34 5403]]
              precision    recall  f1-score   support

         0.0       0.98      0.98      0.98      5560
         1.0       0.98      0.92      0.95      6655
         2.0       0.94      0.95      0.95      5888
         3.0       0.83      0.97      0.90      5819
         4.0       0.96      0.93      0.94      5722
         5.0       0.97      0.87      0.92      5539
         6.0       0.95      0.98      0.96      5858
         7.0       0.94      0.95      0.94      6097
         8.0       0.95      0.91      0.93      5695
         9.0       0.91      0.93      0.92      5813

    accuracy                           0.94     58646
   macro avg       0.94      0.94      0.94     58646
weighted avg       0.94      0.94      0.94     58646
```

4.2. Naive Bayes

```
[[5220    1   11   32    2    1   41    0  251    1]
 [   1 5184  583  238   86   22   85  340   80   36]
 [   9   24 5289  447    4    1    8   52   53    1]
 [   2    1  212 5390    1   33    0  127   31   22]
 [  14    2   44   12 5273    0   32   44   90  211]
 [   9    6   29  103   31 4958   46    2  169  186]
 [  78    7   89    8   15   90 5286    0  285    0]
 [   1   47  175  426   21    1    1 5323   60   42]
 [ 175    5   53  182   23    7   38   13 5112   87]
 [  25    5   62  151  221    4    0   55  184 5106]]
              precision    recall  f1-score   support

         0.0       0.94      0.94      0.94      5560
         1.0       0.98      0.78      0.87      6655
         2.0       0.81      0.90      0.85      5888
         3.0       0.77      0.93      0.84      5819
         4.0       0.93      0.92      0.93      5722
         5.0       0.97      0.90      0.93      5539
         6.0       0.95      0.90      0.93      5858
         7.0       0.89      0.87      0.88      6097
         8.0       0.81      0.90      0.85      5695
         9.0       0.90      0.88      0.89      5813

    accuracy                           0.89     58646
   macro avg       0.90      0.89      0.89     58646
weighted avg       0.90      0.89      0.89     58646
```

4.3. LDA

```
[[5358   10   11   15   19    0   47   17   80    3]
 [   0 6027  222   85    9   22   38  199   31   22]
 [  22   41 5605   12    1    0    4  175   27    1]
 [   1   12   29 5470    1   19    1  247   23   16]
 [  20   71   42    0 5208    0   86    5   29  261]
 [   9   11    6  314    4 5015   50   24   67   39]
 [  77   49   37   15   56   36 5460    0  125    3]
 [   0   58   47    6   58    1    0 5882   22   23]
 [  80   59   38    5   51   29   54   57 4961  361]
 [  34   31    9   91   69    7   16   98   29 5429]]
```

```
              precision    recall  f1-score   support

         0.0       0.96      0.96      0.96      5560
         1.0       0.95      0.91      0.93      6655
         2.0       0.93      0.95      0.94      5888
         3.0       0.91      0.94      0.92      5819
         4.0       0.95      0.91      0.93      5722
         5.0       0.98      0.91      0.94      5539
         6.0       0.95      0.93      0.94      5858
         7.0       0.88      0.96      0.92      6097
         8.0       0.92      0.87      0.89      5695
         9.0       0.88      0.93      0.91      5813

    accuracy                           0.93     58646
   macro avg       0.93      0.93      0.93     58646
weighted avg       0.93      0.93      0.93     58646
```

4.4. Logistic Regression

```
[[5381    5   16   12   15    4   69    6   51    1]
 [   1 5595  116  269  200   74  179   74   78   69]
 [  22   18 5585   89   12    1   33   82   45    1]
 [   4    3   37 5597   16   39    1   74   20   28]
 [  35    8   30    1 5315    2  104   41    9  177]
 [   6   12   23  497   78 4728   50   22   73   50]
 [  87   26    0    1   20   96 5517    0  111    0]
 [   0   41   40  121  165    2    0 5600   17  111]
 [  83   43   47   59   85   46   53   58 5000  221]
 [  55   22    8  143  251    0    4  150   19 5161]]
              precision    recall  f1-score   support

         0.0       0.95      0.97      0.96      5560
         1.0       0.97      0.84      0.90      6655
         2.0       0.95      0.95      0.95      5888
         3.0       0.82      0.96      0.89      5819
         4.0       0.86      0.93      0.89      5722
         5.0       0.95      0.85      0.90      5539
         6.0       0.92      0.94      0.93      5858
         7.0       0.92      0.92      0.92      6097
         8.0       0.92      0.88      0.90      5695
         9.0       0.89      0.89      0.89      5813
```

```
                                                                
    accuracy                           0.91     58646
   macro avg       0.91      0.91      0.91     58646
weighted avg       0.91      0.91      0.91     58646
```

4.5 Perceptron

```
[[5532    1    0    6    0    1   18    1    1    0]
 [  14 6114   46  217   14  176   27   43    2    2]
 [  88   32 5548  137    2    0   16   62    3    0]
 [   5    3   12 5698    0   60    1   28    2   10]
 [ 116   13   46   17 5172    7  108   39    5  199]
 [  21    5    4  129    3 5318   40    1    6   12]
 [ 129    8    5    4    5   57 5648    0    2    0]
 [   2   42   51  157   31    4    0 5796    1   13]
 [ 329   39   45  457   35  225  185   20 4211  149]
 [  89   36   26  115  106   25    3   83    3 5327]]
              precision    recall  f1-score   support

         0.0       0.87      0.99      0.93      5560
         1.0       0.97      0.92      0.94      6655
         2.0       0.96      0.94      0.95      5888
         3.0       0.82      0.98      0.89      5819
         4.0       0.96      0.90      0.93      5722
         5.0       0.91      0.96      0.93      5539
         6.0       0.93      0.96      0.95      5858
         7.0       0.95      0.95      0.95      6097
         8.0       0.99      0.74      0.85      5695
         9.0       0.93      0.92      0.92      5813

    accuracy                           0.93     58646
   macro avg       0.93      0.93      0.93     58646
weighted avg       0.93      0.93      0.93     58646
```

4.6 Melhores combinações

Uma boa combinação de algoritmos seria o Perceptron e o KNN. Como os dados são balanceados, é possível usar a precisão para compará-los. Dessa forma, observando os valores de precisão a partir das matrizes de confusão, esses dois comparadores podem ser bons complementos um do outro.