Aprendizagem de Máquina
Prof. Tiago Buarque A. de Carvalho

Exercícios sobre Avaliação de Classificadores

1. (10 pontos) Qual é o melhor classificador?

Depende do problema e da situação. Não existe um classificador melhor do que os outros de forma geral. Entretanto, certos classificadores se saem melhor que os outros em determinadas situações.

1-NN, 3-NN, 23-NN, Árvore de decisão, SVM, MLP

2. (10 pontos) Qual a diferença entre avaliação qualitativa e quantitativa de classificadores? Cite exemplos de critérios de avaliação qualitativa de classificadores. Cite exemplos de critérios de avaliação quantitativa de classificadores.

Um classificador de avaliação qualitativa possui um grau maior de interpretabilidade, onde a partir de um modelo de regras e talvez uma árvore de decisão, por exemplo, um usuário seja capaz de entender qual foi o modelo que o classificador aprendeu.

Já na avaliação quantitativa, são geradas métricas a partir dos dados inseridos, que revelam informações como a chance de erro/acerto, quanto tempo será gasto para treinar o classificador, quanto tempo o classificador levará para classificar os dados inseridos, etc.

3. (30 pontos) Realize um Holdout com 1-NN (distância Euclidiana), utilizando 70% dos dados para treinamento e o restante (30%) para teste na base de dados Wine archive.ics.uci. edu/ml/datasets/Wine. Você precisa mostrar como calculou cada métrica, não pode utilizar biblioteca que já calcula a métrica diretamente mas pode utilizar biblioteca para o 1-NN e para dividir os dados entre treino e teste. Dica: pode utilizar as seguintes bibliotecas.

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html#sklearn.model_selection.train_test_split

e

https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifhtml#sklearn.neighbors.KNeighborsClassifier

(a) Calcule a matriz de confusão.

```python
# MATRIZ DE CONFUSÃO
def calc_confusion_matrix_n(y_teste, y_pred):

    confusion_matrix_n = []
    matrix_build = []
    list_class_uniq = y_teste.Class.unique()

    # construção de matriz zerada
    for x in range(len(list_class_uniq)):
        for y in range(len(list_class_uniq)):
            matrix_build.append(0)

        confusion_matrix_n.append(matrix_build)
        matrix_build = []

    for x in range(len(y_pred)):
        confusion_matrix_n[y_teste.Class.to_numpy()[x]-1][y_pred[x]-1] += 1

    count = 1

    print("MATRIZ DE CONFUSÃO \n")
    # COLUNA
    print("  1   2   3")

    for x in confusion_matrix_n:
        print(str(count) + ' ' + str(x))
        count += 1

    return confusion_matrix_n
```

```
 (a) Calcule a matriz de confusão:

MATRIZ DE CONFUSÃO

  1   2   3
1 [17, 1, 0]
2 [1, 19, 2]
3 [0, 0, 15]
```

(b) Calcule o Recall por classe.

```python
# RECALL
def calc_recall_class(confusion_matrix_n):
    recall_current = 0
    vp = 0
    fn = 0
    recall = []

    for i in range(len(confusion_matrix_n)):
        for x in range(len(confusion_matrix_n[i])):
            if x == recall_current:
                vp = confusion_matrix_n[i][x]
            else:
                fn += confusion_matrix_n[i][x]

        recall_current += 1

        # FUNÇÃO DO CALCULO DE RECALL
        cal_recall = (vp / (vp + fn))
        fn = 0

        recall.append(cal_recall)

    count = 1

    print ("RECALL \n")
    for K in recall:
        print( count, format(K, '.1f'))
        count += 1

    return recall
```

```
  (b) Calcule o Recall por classe

 RECALL

 1 0.9
 2 0.9
 3 1.0
```

(c) Calcule a Taxa de acerto do classi cador.

```
  (c) Calcule a Taxa de acerto do classificador

 TAXA DE ACERTO

 0.93
```

```python
# TAXA DE ACERTO
def calc_hit_rate(confusion_matrix_n):
    true = 0
    v = 0
    total = 0

    for i in range(len(confusion_matrix_n)):
        for x in range(len(confusion_matrix_n[i])):
            if x == true:
                v += confusion_matrix_n[i][x]
            total += confusion_matrix_n[i][x]

        true += 1

    hit_rate = (v / (total))

    print("TAXA DE ACERTO \n")
    print(format(hit_rate, '.2f'))

    return format(hit_rate, '.2f')
```

(d) Calcule a Precisão por classe.

```python
# PRECISÃO
def calc_precision(confusion_matrix_n):
    current_accuracy = 0
    vp = 0
    fp = 0
    precision = []

    for i in range(len(confusion_matrix_n)):
        for x in range(len(confusion_matrix_n[i])):
            if x == current_accuracy:
                vp = confusion_matrix_n[i][x]
            else:
                fp += confusion_matrix_n[x][i]

        current_accuracy += 1
        calculo = (vp / (vp + fp))
        fp = 0

        precision.append(calculo)

    count = 1
    print ("PRECISÃO \n")
    for y in precision:
        print( count, format(y, '.2f'))
        count += 1

    return precision
```

```
 (d) Calcule a Precisão por classe

PRECISÃO

1 0.94
2 0.95
3 0.88
```

(e) Calcule a Medida-F por classe.

```python
# Medida-F
def calc_measure_f(precision, recall):
    f = []
    for i in range(len(precision)):

        f.append((2*precision[i]*recall[i])/(precision[i]+recall[i]))

    count = 1
    print("MEDIDA F \n")
    for x in f:
        print(count, format(x, '.2f'))
        count += 1

    return f
```

```
 (e) Calcule a Medida-F por classe

MEDIDA F

1 0.94
2 0.90
3 0.94
```

(f) Calcule a Taxa de FP por classe.

```python
# TAXA DE FP
def calc_fp_rate(confusion_matrix_n):
    flag = 0
    fp = 0
    vp = 0
    vp_array = []
    fp_array = []
    fp_rate = []

    for p in range(len(confusion_matrix_n)):
        for t in range(len(confusion_matrix_n)):
            for k in range(len(confusion_matrix_n[t])):
                if t != p and k != p:
                    vp += confusion_matrix_n[k][t]

        vp_array.append(vp)
        vp = 0

    for i in range(len(confusion_matrix_n)):
        flag = 0
        for x in range(len(confusion_matrix_n[i])):
            if i != flag:
                fp += confusion_matrix_n[x][i]
            flag += 1

        fp_array.append(fp)
        fp = 0

    for j in range(len(fp_array)):
        fp_rate.append(fp_array[j] / (vp_array[j] + fp_array[j]))


    count = 1
    print("TAXA DE FP \n")
    for y in fp_rate:
        print( count, format(y, '.2f'))
        count += 1

    return fp_rate
```

```
 (f) Calcule a Taxa de FP por classe

TAXA DE FP

1 0.03
2 0.03
3 0.05
```

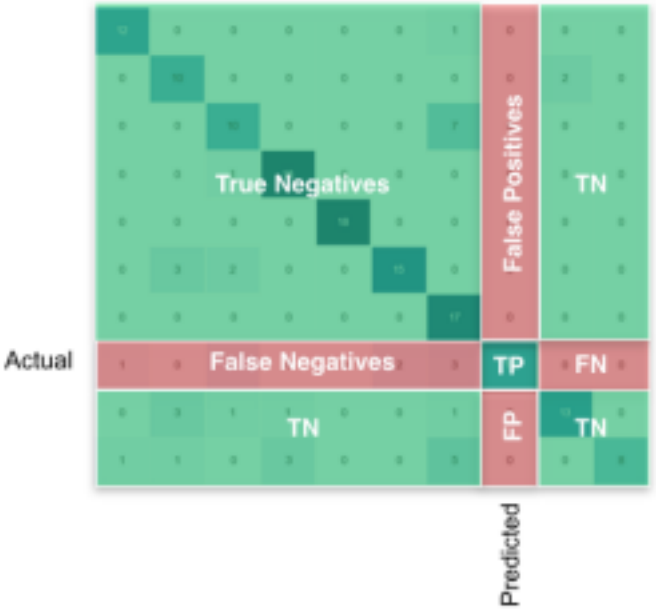

Figura 1: VP (TP), VN (TN), FP e FN na matriz de confusão. As linhas da matriz representam as classes reais e as colunas as classes preditas.

4. (10 pontos) Compare com os resultados da questão anterior com resultados das métricas computados a partir de uma biblioteca que já calcula diretamente estas métricas. Dica: você pode utilizar sklearn.metrics. https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics

(a) Calcule a matriz de confusão.

3.

```
 (a) Calcule a matriz de confusão:

MATRIZ DE CONFUSÃO

   1   2   3
1 [17, 1, 0]
2 [1, 19, 2]
3 [0, 0, 15]
```

4.

```
 (a) Calcule a matriz de confusão:

[[17  1  0]
 [ 1 19  2]
 [ 0  0 15]]
```

(b) Calcule o Recall por classe.

3.

```
 (b) Calcule o Recall por classe

RECALL

1 0.9
2 0.9
3 1.0
```

4.

```
recall

  0.94
  0.86
  1.00
```

(c) Calcule a Taxa de acerto do classi cador.

3.

```
 (c) Calcule a Taxa de acerto do classificador

TAXA DE ACERTO

0.93
```

4.

```
accuracy                          0.93
```

(d) Calcule a Precisão por classe.

3.

```
 (d) Calcule a Precisão por classe

PRECISÃO

1 0.94
2 0.95
3 0.88
```

4.

```
    precision

1       0.94
2       0.95
3       0.88
```

(e) Calcule a Medida-F por classe.

3.

```
 (e) Calcule a Medida-F por classe

MEDIDA F

1 0.94
2 0.90
3 0.94
```

4.

5. (25 pontos) Para cada um dos problemas abaixo, responda:

(a) Entre Precisão e Recall, indique qual métrica de avaliação você acha mais adequada para cada um dos seguintes problemas. A métrica deve ser calculada para a classe positiva.

(b) Quando sua resposta foi Precisão indique o que significa maximizar o Recall, quando sua resposta foi Recall indique o signi caria maximizar a Precisão.

i. Login por impressão digital (ngerprint) em um dispositivo móvel. Nesta aplicação é aconselhável que o usuário consiga logar o maior número possível de vezes, mesmo correndo o risco de uma pequena probabilidade de que outra pessoa consiga se passar por ele. Classe positiva: a impressão digital é do usuário. Classe negativa: a impressão digital não é do usuário.

Nesse caso, a métrica mais adequada seria Recall. Maximizar a precisão possibilitaria que o dispositivo não fosse desbloqueado, em uma considerável quantidade de vezes

ii. Classificar o email como SPAM (Sending and Posting Advertisement in Mass) mas evitando que mensagens importantes sejam enviadas para a lixeira. Isto pode ter como consequência que alguns SPAMs vão chegar à caixa de entrada. Classe positiva: é SPAM. Classe negativa: não é SPAM.

A métrica mais adequada é a Precisão, precisamos de mais precisão, para garantir que as mensagens importantes não sejam classificadas como email, então temos que classificar o maior número possível de spams corretamente. Maximizar o recall, significaria “correr o risco” de classificar um considerável número de mensagens importantes como spam.

iii. Classificar uma fruta saudável ou doente. Escolher o máximo de frutas saudáveis mesmo que alguma fruta doente apareça no meio daquelas selecionadas. Classe positiva: saudável. Classe negativa: doente.

A métrica adequada é o Recall. Maximizar a precisão tornaria possível que muitas frutas saudáveis fossem jogadas fora e queremos o maior número possível de frutas saudáveis.

iv. Detecção de faces, dizer que uma imagem é uma face humana somente quando tiver muita certeza de que é uma face. Deve-se evitar dizer que uma imagem é uma face quando não é de fato. Classe positiva: é uma face. Classe negativa: não é uma face.

A métrica é Precisão. Se maximizar o recall, tornaria possível classificar imagens que não eram face, quando não forem.

Exemplo: Classificação de imagem de face como criminoso procurado (classe positiva) pessoa comum (classe negativa). O sistema deve encontrar o máximo número possível de criminosos procurados mesmo que algumas pessoas inocentes sejam indicadas. Uma pessoa indicada como criminosos procurada será posteriormente investigada por um humano. Exemplo de resposta: Deve-se

maximizar o recall. Maximizar a precisão significaria indicar uma imagem como pertencendo a um criminoso procurado somente quando tivesse muita certeza (o que diminuiria o custo com investigação humana) mas poderia deixar passar mais criminosos procurados do que no caso de maximizar o recall.

6. (15 pontos) A Figura 2 mostra três curvas ROC. Quais justi cativas você teria para:

(a) Escolher o classificado B em detrimento do classificado A.

O classificador B possui uma taxa de verdadeiro positivo maior, fazendo com que possua maior área na curva ROC.

(b) Escolher o classificado B em detrimento do classificado C.

O classificador B tem taxa maior de verdadeiro positivo.

(c) Escolher o classificado C em detrimento do classificado B.

O classificador C, aparenta ter uma taxa menor de falso positivo

2

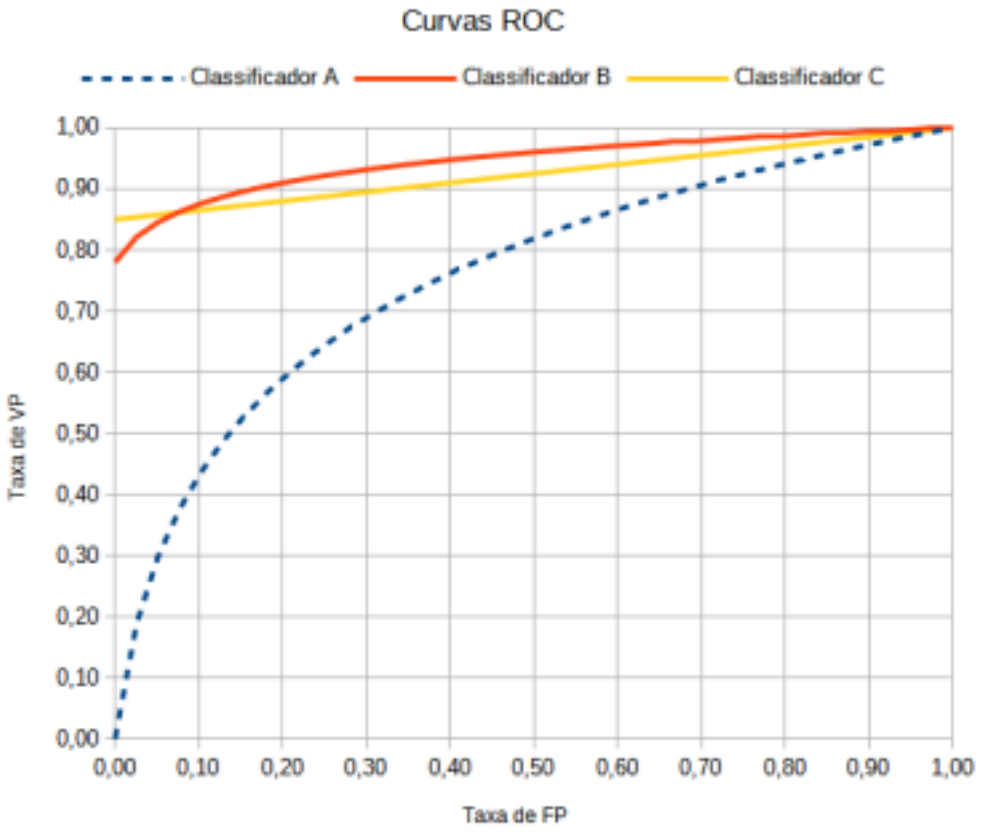

Figura 2: Curvas ROC