Aprendizagem de Máquina
Prof. Tiago Buarque A. de Carvalho
Aluno: Vinícius Santos de Almeida

Exercícios sobre Avaliação de Classificadores

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

Depende, um classificador pode ser definido como melhor para um problema, mas para outro problema ele pode não ser tão bom, também é possível que um classificador seja bom para observar um determinado fator de um problema melhor do que outros, então, para responder qual o melhor classificador, é necessário observar o tipo de problema com o qual está sendo trabalhado, em relação a o que esse classificador é melhor e em que fatores ele é melhor.

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.

Na avaliação qualitativa estamos observando se o classificador avaliado possui determinadas **qualidades**, ou seja, queremos saber se o classificador vai nos mostrar dados legíveis para humanos, com que tipo de dado ele trabalha, qual o formato da entrada e saída, se é possível aumentar facilmente o conjunto de treino, dentre outras qualidades.

Já na avaliação quantitativa, queremos saber como o classificador performa em valores numéricos, desta forma, queremos saber qual a taxa de velocidade para adicionar uma informação no conjunto de treino, com que velocidade ele realiza a classificação dos dados, qual a taxa de erro do classificador, dentre outras métricas quantitativas.

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.

a. Calcule a matriz de confusão.

```
Matriz de confusão:
('1', '2', '3')
1 (17, 0, 1)
2 (4, 12, 2)
3 (3, 9, 6)
```

```python
def matriz_confusao():
    matriz_confusao = {
        CLASSE1: {CLASSE1: 0, CLASSE2: 0, CLASSE3: 0},
        CLASSE2: {CLASSE1: 0, CLASSE2: 0, CLASSE3: 0},
        CLASSE3: {CLASSE1: 0, CLASSE2: 0, CLASSE3: 0},
    }

    for i, value in enumerate(menores):
        if i < 18:
            if value[0] == CLASSE1:
                matriz_confusao[CLASSE1][CLASSE1] += 1
            else:
                matriz_confusao[CLASSE1][value[0]] += 1
        elif i >= 18 and i < 36:
            if value[0] == CLASSE2:
                matriz_confusao[CLASSE2][CLASSE2] += 1
            else:
                matriz_confusao[CLASSE2][value[0]] += 1
        elif i >= 36:
            if value[0] == CLASSE3:
                matriz_confusao[CLASSE3][CLASSE3] += 1
            else:
                matriz_confusao[CLASSE3][value[0]] += 1
    return matriz_confusao
```

```python
print('Matriz de confusão: ')
matriz_confusao = matriz_confusao()
print(tuple(matriz_confusao.keys()))
for key in matriz_confusao:
    print(key, tuple(matriz_confusao[key].values()))
```

Nesse print, menores é uma lista que contém tuplas no formato (classificação,distância) de cada item dos 30% de teste, em cada iteração ele verifica se o intervalo que deveria ser de uma determinada classe foi de fato classificada como tal, caso sim, é adicionado à matriz na linha e coluna da classe, se não, é colocado na linha da classe correta, porém na coluna da classe na qual foi classificada.

b. Calcule o Recall por classe.

```
Recall classe 1: 94.44%
Recall classe 2: 66.67%
Recall classe 3: 33.33%
```

```python
def verdadeiro_positivo(matriz_confusao, classe):
    vp = matriz_confusao[classe][classe]
    return vp

def falso_negativo(matriz_confusao, classe):
    fn = 0
    for key in matriz_confusao[classe]:
        if key != classe:
            fn += matriz_confusao[classe][key]
    return fn

def falso_positivo(matriz_confusao, classe):
    fp = 0
    for key1 in matriz_confusao:
        if key1 != classe:
            for key2 in matriz_confusao[key1]:
                if key2 == classe:
                    fp += matriz_confusao[key1][key2]
    return fp

def recall(classe):
    vp = verdadeiro_positivo(matriz_confusao, classe)
    fn = falso_negativo(matriz_confusao, classe)
    recall = vp / (vp + fn)
    return recall
```

* Função falso_positivo não usada nessa letra da questão.

```python
recall_classe1 = recall(CLASSE1)
print(f'Recall classe 1: {round(recall_classe1 * 100, 2)}%')
recall_classe2 = recall(CLASSE2)
print(f'Recall classe 2: {round(recall_classe2 * 100, 2)}%')
recall_classe3 = recall(CLASSE3)
print(f'Recall classe 3: {round(recall_classe3 * 100, 2)}%')
```

c. Calcule a taxa de acerto do classificador.

```
Acertos: 64.81%
```

```python
acertos = verdadeiro_positivo(matriz_confusao, CLASSE1) \
    + verdadeiro_positivo(matriz_confusao, CLASSE2) \
    + verdadeiro_positivo(matriz_confusao, CLASSE3)
print(f"Acertos: {round(acertos * 100 / len(menores), 2)}%")
```

d. Calcule a Precisão por classe.

```
Precisão classe 1: 70.83%
Precisão classe 2: 57.14%
Precisão classe 3: 66.67%
```

```python
def precisao(classe):
    vp = verdadeiro_positivo(matriz_confusao, classe)
    fp = falso_positivo(matriz_confusao, classe)
    precisao = vp / (vp + fp)
    return precisao

```

```python
precisao_classe1 = precisao(CLASSE1)
print(f'Precisão classe 1: {round(precisao_classe1 * 100, 2)}%')
precisao_classe2 = precisao(CLASSE2)
print(f'Precisão classe 2: {round(precisao_classe2 * 100, 2)}%')
precisao_classe3 = precisao(CLASSE3)
print(f'Precisão classe 3: {round(precisao_classe3 * 100, 2)}%')
```

* Funções verdadeiro_positivo e falso_positivo já mostradas na letra b.

e. Calcule a Medida-F por classe.

```
Medida-F classe 1: 80.95%
Medida-F classe 2: 61.54%
Medida-F classe 3: 44.44%
```

```python
def medida_f(classe):
    p = precisao(classe)
    r = recall(classe)
    f = 2 * p * r / (p + r)
    return f
```

```python
f_classe1 = medida_f(CLASSE1)
print(f'Medida-F classe 1: {round(f_classe1 * 100, 2)}%')
f_classe2 = medida_f(CLASSE2)
print(f'Medida-F classe 2: {round(f_classe2 * 100, 2)}%')
f_classe3 = medida_f(CLASSE3)
print(f'Medida-F classe 3: {round(f_classe3 * 100, 2)}%')
```

* Funções precisao e recall já mostradas nas letras b e d.

f. Calcule a Taxa de FP por classe.

```
Taxa VN classe 1: 19.44%
Taxa VN classe 2: 25.0%
Taxa VN classe 3: 8.33%
```

```python
def verdadeiro_negativo(matriz_confusao, classe):
    vn = 0
    for key1 in matriz_confusao:
        if key1 != classe:
            for key2 in matriz_confusao[key1]:
                if key2 != classe:
                    vn += matriz_confusao[key1][key2]
    return vn
```

```python
def taxa_fp(classe):
    fp = falso_positivo(matriz_confusao, classe)
    vn = verdadeiro_negativo(matriz_confusao, classe)
    taxa_fp = fp / (fp + vn)
    return taxa_fp
```

```python
t_vn_classe1 = taxa_fp(CLASSE1)
print(f'Taxa VN classe 1: {round(t_vn_classe1 * 100, 2)}%')
t_vn_classe2 = taxa_fp(CLASSE2)
print(f'Taxa VN classe 2: {round(t_vn_classe2 * 100, 2)}%')
t_vn_classe3 = taxa_fp(CLASSE3)
print(f'Taxa VN classe 3: {round(t_vn_classe3 * 100, 2)}%')
```

* Função falso_positivo já apresentada na letra b.

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.

```
Matriz de confusão:
('1', '2', '3')
1 (16, 1, 1)
2 (1, 15, 7)
3 (2, 4, 7)
```

```
('1', '2', '3')
Precisão: [0.84210526 0.75       0.46666667]
Recall: [0.88888889 0.65217391 0.53846154]
Métrica-F: [0.86486486 0.69767442 0.5       ]
```

```
Acertos: 0.7037037037037037
```

```python
    def calcular_matriz_confusao(self):

        predicts = self.knn.predict(self.x_teste)
        self.matriz_confusao = confusion_matrix(self.y_teste, predicts)
```

```python
def metrics(self):
    predicts = self.knn.predict(self.x_teste)
    return score(self.y_teste, predicts)
```

```python
def acertos(self):
    return self.knn.score(self.x_teste, self.y_teste)
```

```python
precision, recall, fscore, support = wine.metrics()
print((CLASSE1, CLASSE2, CLASSE3))
print(f'Precisão: {precision}')
print(f'Recall: {recall}')
print(f'Métrica-F: {fscore}')
print(f'Taxa VN: {support}')
print(f'Acertos: {wine.acertos()}')
```

É interessante notar que na maioria das vezes a biblioteca performa melhor em todos os sentidos, o que se dá possivelmente pela forma como ela separa a base em teste e treino, que parece ser aleatória, diferente da minha implementação que separa sempre os mesmos elementos.
Em relação à taxa de FP, considerar o respondido na questão 3, letra F.

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

i. a. Nessa aplicação é mais indicado um classificador que tenha um Recall maior, dado que a aplicação requer que o máximo de tentativas do usuário real (classe positiva) sejam classificadas corretamente, ou seja, que haja o menor número de falso negativos possíveis, mesmo que isso aumente a chances de falso positivos, que permitiria que outros tenha acesso ao dispositivo (classe negativa).

i. b. Ao maximizar a precisão nessa aplicação, isso significaria que haveria mais proteção no dispositivo, em contrapartida, aumentaria as chances do usuário do dispositivo ser atribuído à classe negativa, devido a possível baixa do recall consequente da maximização da precisão.

ii.a. Nessa aplicação é mais indicada a métrica Precisão, já que é esperado que o menor número de emails não SPAM sejam classificados como SPAM, ou seja, diminuir o número de falso positivos.

iii. a. Esse é mais um caso onde é indicado Recall, isso porque é mais importante que todas as frutas saudáveis sejam classificadas como tal, mesmo que algumas frutas doentes se passem por saudáveis, ou seja, menores taxas de falso negativos.

iii. b. Maximizar a precisão nesse caso significaria que a classe positiva teria menos falso positivos, ou seja, haveria mais garantia de que as frutas saudáveis são de fato saudáveis, mesmo que algumas frutas saudáveis sejam classificadas como doentes no processo.

iv. a. Nessa aplicação é mais indicada métrica Precisão, isso daria mais garantia que as imagens classificadas na classe positiva de fato são de faces, ou seja, haveria um número baixíssimo de falso positivos.

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

a. Preferimos o B em detrimento do A, pois é desejável que a taxa de VP seja o maior possível e a taxa de FP seja o menor possível, e analisando os classificadores A e B, nota-se que com as mesmas taxa de FP ele consegue atingir taxas muito maiores de VP, além disso, o classificador B ainda consegue uma maior área ROC, o que também é mais desejável.
b. Preferimos o B invés do C, porque o classificador B consegue atingir melhores taxas de VP nas mesmas taxas de FP do classificador C, na maioria dos casos, e ainda consegue maior área ROC, o que também é desejável.
c. Escolheríamos o classificador C em detrimento de B apenas quando priorizamos taxas de FP muito baixas (menores que 0,10), a ponto de tornar o classificador C mais vantajoso por oferecer taxas de VP maiores nesse intervalo menor que 0,10, mesmo que a área da curva ROC seja menor que no classificador B.