PRIMEIRA LISTA DE EXERCÍCIOS E TRABALHOS

DISCENTES:
FERNANDO LUCAS SOUSA SILVA
TEÓPHILO VITOR DE CARVALHO CLEMENTE

DOCENTE:
ADRIÃO DUARTE DÓRIA NETO

NATAL/RN
2022

1. INTRODUÇÃO

O presente trabalho visa apresentar a resolução dos exercícios propostos pelo professor, como também os trabalhos de pesquisa que foram desenvolvidos para apresentação em sala de aula com o objetivo de reforçar o aprendizado adquirido até então na matéria.

2. METODOLOGIA

Para o desenvolvimento deste trabalho utilizamos de pesquisas no material didático disponibilizado pelo professor e em sites da internet, a partir destes, desenvolvemos a resolução dos exercícios em questão seja a mão ou código, do mesmo modo foi feito para os trabalhos em questão. Para melhor compreensão este relatório foi dividido em partes que apresentam o conteúdo produzido.

3. RESULTADOS

Aqui serão apresentadas as resoluções dos exercícios da lista como ordenado no arquivo descritivo pelo professor.

3.1 Questão 1

Fonte: Material de aula.

Registro	Tem casa própria	Estado Civil	Possui Carro	Rendimentos	Bom Pagador
1	Sim	Solteiro	Sim	Alto	Não
2	Não	Casado	Sim	Médio	Não
3	Não	Solteiro	Não	Baixo	Não
4	Sim	Casado	Sim	Alto	Não
5	Não	Divorciado	Não	Médio	Sim
6	Não	Casado	Não	Baixo	Não
7	Sim	Divorciado	Sim	Alto	Sim
8	Não	Solteiro	Sim	Médio	Sim
9	Não	Casado	Sim	Baixo	Não
10	Não	Solteiro	Não	Médio	Sim
11	Sim	Divorciado	Não	Médio	Não
12	Não	Divorciado	Sim	Alto	?
13	Sim	Solteiro	Não	Médio	?

Figura 1 - Tabela de dados.

a) No caso em questão deseja-se saber se um indivíduo é Bom Pagador (BP), dado que ele: Não tem Casa Própria (TCP), seu Estado Civil (EC) é Divorciado, possui Carro (TC) e possui Rendimento (R) Alto. Dessa forma, calcula-se:

A probabilidade dele ser ou não um bom pagador:

P(BP = Sim) = $\frac{4}{11} = 0,36$

P(BP = Não) = $\frac{7}{11} \simeq 0,64$

P(TCP = Não) = $\frac{7}{11} \simeq 0,64$

P(EC = Divorciado) = $\frac{3}{11} = 0,27$

P(TC = Sim) = $\frac{6}{11} \simeq 0,54$

P(R = Alto) = $\frac{3}{11} = 0,27$

A probabilidade das condições explicitadas, levando-se em consideração os estados de ser Bom Pagador:

P(R = Alto | BP = Sim) = $\frac{1}{4} = 0,25$

P(EC = Divorciado | BP = Sim) = $\frac{2}{4} = 0,5$

P(TCP = Não | BP = Sim) = $\frac{3}{4} = 0,75$

P(TC = Sim | BP = Sim) = $\frac{2}{4} = 0,5$

A probabilidade dele ser Bom Pagador, levando-se em consideração todas as condições:

$$\frac{P(TCP = Não \mid BP = Sim) * P(EC = Divorciado \mid BP = Sim) * P(TC = Sim \mid BP = Sim) * P(R = Alto \mid BP = Sim) * (BP = Sim)}{P(R = Alto) * P(TCP = Não) * P(EC = Divorciado) * P(TC = Sim)} = 0,6602$$

A probabilidade das condições explicitadas, levando-se em consideração os estados de não ser Bom Pagador:

P(R = Alto | BP = Não) = $\frac{2}{7} = 0,29$

P(EC = Divorciado | BP = Não) = $\frac{1}{7} = 0,14$

P(TCP = Não | BP = Não) = $\frac{4}{7} = 0,57$

P(TC = Sim | BP = Não) = $\frac{4}{7} = 0,57$

A probabilidade dele não ser Bom Pagador, levando-se em consideração todas as condições:

$$\frac{P(TCP = Não \mid BP = Não) * P(EC = Divorciado \mid BP = Não) * P(TC = Sim \mid BP = Não) * P(R = Alto \mid BP = Não) * (BP = Não)}{P(R = Alto) * P(TCP = Não) * P(EC = Divorciado) * P(TC = Sim)} = 0,3285$$

R: Portanto, como 0,6602 é maior que 0,3285, conclui-se que o indivíduo 12 é um Bom Pagador.

b) No caso em questão deseja-se saber se um indivíduo é Bom Pagador (BP), dado que ele: Tem Casa Própria (TCP), seu Estado Civil (EC) é Solteiro, não possui Carro (TC) e possui Rendimento (R) Médio . Dessa forma, calcula-se

A probabilidade dele ser ou não um bom pagador:

P(BP = Sim) = $\frac{4}{11} = 0,36$

P(BP = Não) = $\frac{7}{11} \simeq 0,64$

P(TCP = Sim) = $\frac{4}{11} \simeq 0,36$

P(EC = Solteiro) = $\frac{4}{11} = 0,36$

P(TC = Não) = $\frac{5}{11} \simeq 0,45$

P(R = Médio) = $\frac{5}{11} = 0,45$

A probabilidade das condições explicitadas, levando-se em consideração os estados de ser Bom Pagador:

P(R = Médio | BP = Sim) = $\frac{3}{4} = 0,75$

P(EC = Solteiro | BP = Sim) = $\frac{2}{4} = 0,5$

P(TCP = Sim | BP = Sim) = $\frac{1}{4} = 0,25$

P(TC = Não | BP = Sim) = $\frac{2}{4} = 0,5$

A probabilidade dele ser Bom Pagador, levando-se em consideração todas as condições:

$$\frac{P(TCP = Sim \mid BP = Sim) * P(EC = Solteiro \mid BP = Sim) * P(TC = Não \mid BP = Sim) * P(R = Médio \mid BP = Sim) * (BP = Sim)}{P(R = Médio) * P(TCP = Sim) * P(EC = Solteiro) * P(TC = Não)} = 0,6239$$

A probabilidade das condições explicitadas, levando-se em consideração os estados de não ser Bom Pagador:

P(R = Médio | BP = Não) = $\frac{2}{7} = 0,29$

P(EC = Solteiro | BP = Não) = $\frac{2}{7} = 0,29$

P(TCP = Sim | BP = Não) = $\frac{3}{7} = 0,43$

P(TC = Não | BP = Não) = $\frac{3}{7} = 0,43$

A probabilidade dele não ser Bom Pagador, levando-se em consideração todas as condições:

$$\frac{P(TCP = Sim \mid BP = Não) * P(EC = Solteiro \mid BP = Não) * P(TC = Não \mid BP = Não) * P(R = Médio \mid BP = Não) * (BP = Não)}{P(R = Médio) * P(TCP = Sim) * P(EC = Solteiro) * P(TC = Não)} = 0,3492$$

R: Portanto, como 0,6239 é maior que 0,3492, conclui-se que o indivíduo 13 é um Bom Pagador.

3.3 Questão 3

a) Gráfico da rede bayesiana:

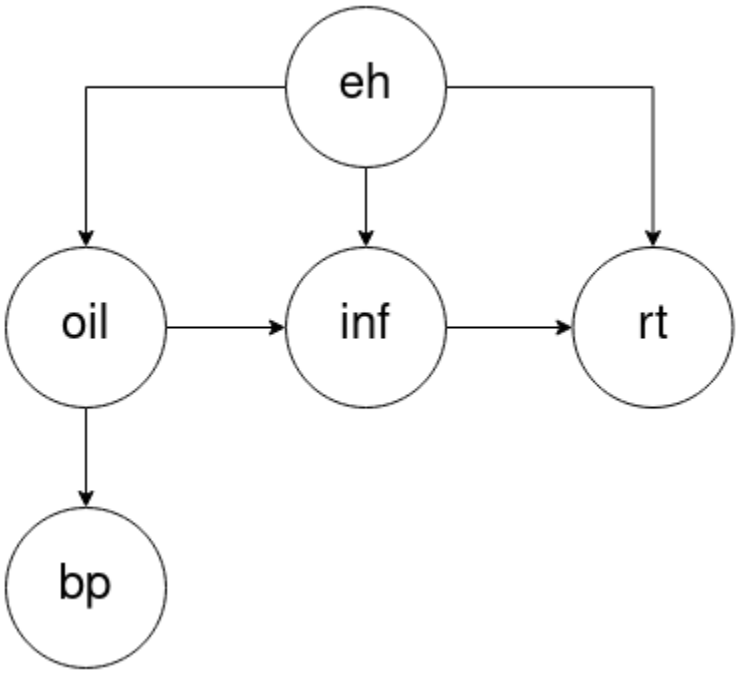

Figura 2 - Rede Bayesiana.

b)

Para analisarmos a probabilidade de a inflação ser alta teremos a seguinte a expressão:

$$P(inf = h|bp = n, rt = h) = \frac{P(inf=h,\, bp=n,\, rt=h)}{P(bp=n,\, rt=hi)} = \frac{P(inf=h|eh,oil)\cdot P(bp=n|oil)\cdot P(\, rt=h|eh,inf=h)}{P(bp=n,\, rt=hi)}$$

Para resolver a expressão final vamos executá-la por partes, primeiro vamos calcular P(inf = h | eh, oil) pela seguinte expressão:

$$P(inf = h|eh, oil) = [\, P(inf = h|eh = l,\, oil = l) \cdot P(eh = l) \cdot P(oil = l|eh = l)\,] + [\, P(inf = h|eh = h,\, oil = h) \cdot P(eh = h) \cdot P(oil = h|eh = h)\,] + [\, P(inf = h|eh = l,\, oil = h) \cdot P(eh = l) \cdot P(oil = h|eh = l)\,] + [\, P(inf = h|eh = h,\, oil = l) \cdot P(eh = h) \cdot P(oil = l|eh = h)\,]$$

Desenvolvendo as probabilidades obtemos:

$$P(inf = h|eh, oil) = 0.6511$$

Agora vamos calcular P(bp=n| oil) pela seguinte expressão:

$$P(bp = n|oil) = [\, P(bp = n|oil = h) \cdot [P(oil = h|eh = l) \cdot P(eh = l) + P(oil = h|eh = h) \cdot P(eh = h)]\,] + [\, P(bp = n|oil = l) \cdot [P(oil = l|eh = l) \cdot P(eh = l) + P(oil = l|eh = h) \cdot P(eh = h)]\,]$$

Desenvolvendo as probabilidades obtemos:

$$P(bp = n|oil) = 0.2065$$

Agora vamos calcular P(rt=h | eh, inf=h) pela seguinte expressão:

$$P(\, rt = h|eh, inf = h) = [\, P(rt = h|inf = h,\, eh = l) \cdot P(inf = h|eh = l,\, oil) \cdot P(eh = l)\,] + [\, P(rt = h|inf = h,\, eh = h) \cdot P(inf = h|eh = h,\, oil) \cdot P(eh = h)\,]$$

Desenvolvendo as probabilidades obtemos:

$$P(\, rt = h|eh, inf = h) = 0.1672$$

Dado que calculamos já os componentes do numerador da nossa expressão final agora vamos desenvolver o denominador, então agora vamos calcular P(bp=n | rt=hi) pela seguinte expressão:

$$P(bp = n, rt = hi) = P(bp = n|oil) \cdot P(rt = h|inf, eh)$$

Temos que $P(bp = n|oil) = 0.2065$ e $P(rt = h|inf, eh) = 0.4185$ desse como temos que:

$$P(bp = n, rt = hi) = 0.0864$$

Com isso temos todas as probabilidades necessárias já calculadas e nos basta agora substituir na primeira equação que vimos no início da questão para saber a probabilidade da inflação ser alta. Assim, substituindo todos os valores temos:

$$P(inf = h|bp = n, rt = h) = \frac{0.6511 \cdot 0.2065 \cdot 0.1672}{0.0864} = 0.26$$

Ou seja, a probabilidade da inflação ser alta é **0.26**.

3.5 Questão 5

Primeiramente devemos falar da Matriz de Confusão, ela é uma tabela que nos indica os erros e acertos do modelo, fazendo assim um comparativo ao resultado esperado, ela é constituída por:

- Verdadeiro positivo: é a classificação correta quando a classe é positiva.
- Falso positivo: O modelo prevê a classe positiva quando o correto era a negativa.
- Verdadeiro negativo: é a classificação correta quando a classe é negativa.
- Falso negativo: O modelo prevê a classe negativa quando o correto era a positiva.

A seguir uma imagem ilustrando a composição da Matriz de Confusão:

Fonte: ResearchGate.

		Valor Verdadeiro	
		Classe Positiva	Classe Negativa
Valor previsto	Classe Positiva	**VP** Verdadeiro Positivo	**FP** Falso Positivo
	Classe Negativa	**FN** Falso Negativo	**VN** Verdadeiro Negativo

Figura 3 - Matriz de Confusão.

Partindo do entendimento do modelo a partir da matriz temos que a **Acurácia** indicará a performance geral do modelo, se ele tem ou não um bom desempenho dentre todas as classificações, baseando-se em quantas vezes o modelo classificou corretamente, a seguir a sua fórmula:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

Já a **Precisão** avaliará dentro todas as classificações que foram dadas como positivas, aquelas que realmente estão corretas, na prática ela evita que sejam computados falsos positivos, visto que, sua computação como uma coisa positiva quando não é pode ser bastante prejudicial em alguns casos, a seguir a sua fórmula:

$$Precisão = \frac{VP}{VP + FP}$$

Não foi pedido, mas para fins de entendermos a fórmula mais a frente temos o **Recall**, ele avaliará dentre todas as situações de classe Positivo como valor esperado, quantas estão corretas, na prática ele é contrário a precisão no quesito que ele evita a computação de falsos negativos, pois neste caso esses seriam mais prejudiciais, a seguir sua fórmula:

$$Recall = \frac{VP}{VP + FN}$$

Por último temos o **F-Score**, ele é uma média harmônica entre a Precisão e Recall, na prática ele irá mensurar o nível de ambos, Precisão e Recall em um único cálculo. Ao avaliarmos ele e obtermos um valor baixo sabemos que um dos dois elementos que o compõem está baixo para assim fazermos a nossa avaliação específica de cada um se for preciso, a seguir o seu cálculo:

$$F - Score = 2 * \frac{Precisão * Recall}{Precisão + Recall}$$

4. TRABALHOS

Aqui serão apresentados os trabalhos como ordenado no arquivo descritivo pelo professor.

4.1 Naive Bayes - Filtro anti-spam

Esse trabalho visa realizar uma pesquisa em torno do algoritmo de Naive Bayes aplicado para a utilização em filtro de e-mails com spam. Com isso, após realizada a pesquisa se torna possível a apresentação de um projeto realizado em Python para a análise de um arquivo com mensagens consideradas spam e outras normais.

Como dito anteriormente, desenvolvido em Python, inicialmente utilizaremos a importação das bibliotecas necessárias para o projeto.

```
import nltk
#módulo para trabalhar com stopwords, que são palavras que podem ser filtradas
#do texto, como "the", "is", "are"
from nltk.corpus import stopwords
#funções que trabalham com operações envolvendo cadeias de palavras
import string
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report,confusion_matrix
```

Figura 4 - Importação de bibliotecas utilizadas.

O próximo passo é a criação de uma função que será responsável por interpretar o texto da mensagem de e-mail.

```
def processaTexto(texto):
    #remove pontuação caractere a caractere
    nopunc = [char for char in texto if char not in string.punctuation]
    #junta os caracteres em palavras novamente
    nopunc = ''.join(nopunc)

    #nopunc.split() separa cada frase em palavras, retirando as que estão dentro de stopwords
    #word.lower() torna todas as letras minúsculas
    #essa linha remove as stopwords, retornando apenas as palavras relevantes para a análise
    cleanWords = [word for word in nopunc.split() if word.lower() not in stopwords.words('english')]

    return cleanWords
```

Figura 5 - Função que interpreta o texto das mensagens.

Assim, continuando o desenvolvimento, é necessário uma base de dados, nesse caso foi utilizado um arquivo com possíveis mensagens de e-mail, já classificadas previamente como “spam” ou “ham”, sendo assim, nosso objeto de estudo.

```
data = pd.read_csv('./spam.csv', encoding='latin-1')
data.head(n=20)
```

Figura 6 - Leitura dos dados para o Python e apresentação dos 20 primeiros em tabela.

Portanto, o próximo e último passo é realizar o treinamento e testes com os dados fornecidos, além de aplicar um modelo para realização do mesmo. O pipeline realiza uma forma de modelo de treinamento, utilizando de 3 métodos: a contagem de palavras semelhantes em mensagens de “spam” e “ham”; a conversão desses valores semelhantes em valores válidos, separando os que são semelhantes em ambos; aplicação do treinamento baseada no algoritmo de Naive Bayes.

```
#divide os conjuntos em treinamento e teste
#msg_train e msg_test são os conjuntos de treinamento e teste da base de dados de mensagens
#class_train e class_test são, respectivamente, os rótulos dos sets de teste e treinamento
#por parâmetro são passados os conjuntos de mensagens e o conjunto de rótulos (tipo), bem como o tamanho da base de teste
msg_train, msg_test, class_train, class_test = train_test_split(messages['mensagem'], messages['tipo'], test_size=0.1)

#pipeline de transformação com estimador. sequencialmente aplica uma lista de transformações
#a primeira converte documentos de texto em uma matriz com contagem de tokens, ou seja, conta a ocorrência de cada palavra no vocabulário
#a segunda normaliza a contagem (term frequency times inverse document frequency)
#este escala para baixo o impacto de tokens que ocorrem frequentemente e que sejam
#empiricamente menos informativos do que features que ocorrem em uma pequena fração do treinamento
#a última linha treina esses vetores no classificador naive bayes
pipeline = Pipeline([
    ('bow',CountVectorizer(analyzer=processaTexto)), # converts strings to integer counts
    ('tfidf',TfidfTransformer()), # converts integer counts to weighted TF-IDF scores
    ('classifier',MultinomialNB()) # train on TF-IDF vectors with Naive Bayes classifier
])

#treina o modelo
pipeline.fit(msg_train,class_train)

#testa o modelo
predictions = pipeline.predict(msg_test)

#mostra os resultados
print(classification_report(class_test,predictions))
```

Figura 7 - Aplicação do treinamento.

	precision	recall	f1-score
ham	0.96	1.00	0.98
spam	1.00	0.71	0.83
accuracy			0.96
macro avg	0.98	0.85	0.90
weighted avg	0.96	0.96	0.96

Figura 8 - Tabela final de classificação.

4.2 Random Forest - Autenticidade de cédulas

O algoritmo de floresta aleatória é utilizado combinando vários algoritmos semelhantes, ou seja, vários algoritmos de árvores de decisão, resultando em uma floresta de árvores, daí o nome "Random Forest". Assim, utilizando de bibliotecas Python que simulam o algoritmo de floresta aleatória será apresentado um projeto que, a partir de 4 parâmetros, verifica se cédulas são verdadeiras ou falsas.

```
import pandas as pd
import numpy as np

[20] dataset = pd.read_csv("./bill_authentication.csv")

[21] dataset.head()
```

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.6661	-2.8073	-0.44699	0
1	4.54590	8.1674	-2.4586	-1.46210	0
2	3.86600	-2.6383	1.9242	0.10645	0
3	3.45660	9.5228	-4.0112	-3.59440	0
4	0.32924	-4.4552	4.5718	-0.98880	0

Figura 9 - Importação de bibliotecas, leitura do arquivo e apresentação dos dados.

Com os dados é possível realizar os processos de preparação para o treinamento, dividindo em atributos e títulos inicialmente e, após isso, separando em conjuntos de treinamento e teste.

```
X = dataset.iloc[:, 0:4].values
y = dataset.iloc[:, 4].values

[23] from sklearn.model_selection import train_test_split

     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Figura 10 - Divisão dos conjuntos de treinamento e teste.

Com isso, já é possível realizar o treinamento com o algoritmo de random forest para resolução desse problema de classificação. Assim, temos:

```
from sklearn.ensemble import RandomForestClassifier

classifier = RandomForestClassifier(n_estimators=20, random_state=0)
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
```

Figura 11 - Treinamento com 20 árvores.

Por fim, para finalizar a análise e classificação, é importante que seja realizada a avaliação das informações já obtidas anteriormente, com os conceitos de acurácia, precisão, recall e F1-score.

```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
print("Matriz de confusão:")
print(confusion_matrix(y_test,y_pred))
print("\n", classification_report(y_test,y_pred))
#print(accuracy_score(y_test, y_pred))
```

```
Matriz de confusão:
[[155   2]
 [  1 117]]

              precision    recall  f1-score   support

           0       0.99      0.99      0.99       157
           1       0.98      0.99      0.99       118

    accuracy                           0.99       275
   macro avg       0.99      0.99      0.99       275
weighted avg       0.99      0.99      0.99       275
```

Figura 12 - Apresentação final e classificação.

5. CONCLUSÕES

A partir do desenvolvimento e resolução das questões foi possível agregar conhecimento ao adquirido em sala e entender as diversas aplicações que os algoritmos de Naive Bayes e Random Forest têm para a resolução de problemas e construção de soluções, com isso, estendendo como de fato se dá sua aplicabilidade em cenários diversos e seus benefícios.

6. REFERÊNCIAS

[1] Material didático Professor Dr. Adrião Duarte.

[2] VARGAS, Pablo. **Naive Bayes & SVM Spam Filtering.** 2017. Disponível em: https://www.kaggle.com/code/pablovargas/naive-bayes-svm-spam-filtering.

[3] LATTARI, Lucas. **Implementando um filtro anti-spam usando Naive Bayes.** 2018. Disponível em: https://github.com/lucaslattari/UniversoDiscreto/tree/master/Naive%20Bayes.

[4] MALIK, Usman. **Algoritmo de floresta aleatória com Python e Scikit-Learn.** Disponível em: https://stackabuse.com/random-forest-algorithm-with-python-and-scikit-learn/.