UNIVERSIDADE CATÓLICA DE PELOTAS

Algoritmos de Aprendizado Supervisionado Implementação

ENGENHARIA DE COMPUTAÇÃO

INTELIGÊNCIA ARTIFICIAL

HELENA GARCIA TAVARES

1. Descrição completa do banco de dados escolhido

A base escolhida para o trabalho foi retirada da plataforma *Keagle* e se trata *de* um conjunto de informações que tem como objetivo prever diagnosticamente casos de diabetes entre mulheres com pelo menos 21 anos com origem indígena Pima. As variáveis preditoras foram:

Pregnancies: número de gestações que teve

Glucose: nível de glicose

BloodPressure: pressão arterial

SkinThickness: espessura da pele

Insulin: nível de insulina

BMI: IMC (índice de massa corporal)

DiabetesPedigreeFunction*:* tendência ao desenvolvimento de diabetes

Age: idade

Outcome: probabilidade de ter diabetes ou não ter (desfecho)

2. Explicação dos algoritmos de aprendizado de máquina escolhidos

- *Logistic Regression*: é baseada em uma equação matemática que procura estimar a probabilidade de ocorrência de um determinado evento a partir de uma ou mais variáveis chamadas variáveis independentes.

- *Random Forest Classifier:* é baseada no algoritmo de *Decision Tree.* Gera um conjunto de B árvores com a utilização de diferentes conjuntos de X variáveis independentes na construção de cada uma das árvores. Podemos análogamente dizer que: a utilização desta técnica para a tomada de decisão corresponde à síntese da opinião de diversos indivíduos com diferentes fontes de informação sobre o problema em questão.

- *K-Neighbors Classifier:* é uma das técnicas mais simples e não precisa de ajuste no modelo, como nas citadas acima. Para cada dado analisado, o algoritmo: calcula distância dele até os vizinhos, encontra K vizinhos mais próximos e classifica de acordo com os K vizinhos. Esse valor do trajeto pode se dar através do cálculo da Distância Euclidiana, Distância de Hamming, Distância Manhattan ou Distância de Markowski.

3. Explicação do conceito e justificativa dos aperfeiçoamentos escolhidos

O pré-processamento dos dados consiste em remodelar ou reorganizar alguns dados do banco, antes de aplicar o algoritmo de aprendizado, com o objetivo de obter um melhor desempenho e resultado. As técnicas escolhidas foram:

- **Exclusão de alguns atributos.** Nesse caso, foi escolhido o atributo da coluna *SkinThickness,* por ser o atributo de menor relevância em relação aos demais contidos no banco e por apresentar bastante valores irrelevantes (zeros).

- Tratamento de dados faltantes. Já nesse caso, o critério foi excluir as linhas, onde a coluna com maior número de zeros, tivessem o valor zero. Linhas com valor zero foram consideradas com dados faltantes, e por isso excluídas do processamento dos dados.

4. Explicação dos códigos em Python para criação dos modelos

Importação das bibliotecas necessárias para o desenvolvimento do algoritmo

```
1  import pandas as pd
2  from sklearn.linear_model import LogisticRegression
3  from sklearn.ensemble import RandomForestClassifier
4  from sklearn.neighbors import KNeighborsClassifier
5  from sklearn.model_selection import train_test_split
6  from sklearn.metrics import accuracy_score
```

Carregamento do banco de dados escolhido

```
9  data = pd.read_csv('diabetes.csv', encoding='utf-8', delimiter=',',usecols=
10                    ['Pregnancies', 'Glucose', 'BloodPressure',
11                    'SkinThickness', 'Insulin', 'BMI',
12                    'DiabetesPedigreeFunction', 'Age','Outcome'])
```

4. Explicação dos códigos em Python para criação dos modelos

Separação das variáveis preditoras, da variável de desfecho e atribuição, respectivamente, à variável X e y

```
15   # separa as variáveis preditoras do desfecho
16   y = data['Outcome']
17   # todas as colunas, exceto a Outcome e grava em x
18   X = data.drop('Outcome', axis = 1)
```

Definindo os dados de treino e de teste de acordo com o pedido (30% teste e 70% treino)

```
21   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

Aplicando as funções relativas aos algoritmos escolhidos aos conjuntos de dados de treinamento

```
23   model1 = LogisticRegression()
24   model1.fit(X_train, y_train)
25
26   model2 = RandomForestClassifier()
27   model2.fit(X_train, y_train)
28
29   model3 = KNeighborsClassifier()
30   model3.fit(X_train, y_train)
```

4. Explicação dos códigos em Python para criação dos modelos

Avaliação dos modelos, comparando com os dados de teste

```
34    yhat1 = model1.predict(X_test)
35    yhat2 = model2.predict(X_test)
36    yhat3 = model3.predict(X_test)
```

Cálculo da acurácia de cada modelo e o print dos resultados em percentual

```
41    accuracy1 = accuracy_score(y_test, yhat1)
42    print('Accuracy LogisticRegression: %.2f' % (accuracy1*100))
43
44    accuracy2 = accuracy_score(y_test, yhat2)
45    print('Accuracy RandomForestClassifier: %.2f' % (accuracy2*100))
46
47    accuracy3 = accuracy_score(y_test, yhat3)
48    print('Accuracy KNeighborsClassifier: %.2f' % (accuracy3*100))
```

4. Explicação dos códigos em Python para criação dos modelos

Funções para aplicar as técnicas de pré-processamento escolhidas:

- Exclusão de atributos

```
15    data.pop('SkinThickness')
```

- Tratamento de Dados Faltantes

```
18    indexNames = data[data['Insulin'] == 0].index
19    data.drop(indexNames , inplace=True)
```

5. Discussão sobre o desempenho atingido pelos modelos com e sem o pré-processamento dos dados

Resultados:*

- sem pré-processamento

```
Accuracy LogisticRegression: 77.92
Accuracy RandomForestClassifier: 75.76
Accuracy KNeighborsClassifier: 70.13
```

- com exclusão de atributos

```
Accuracy LogisticRegression: 78.35
Accuracy RandomForestClassifier: 76.62
Accuracy KNeighborsClassifier: 72.29
```

- com tratamento de dados faltantes

```
Accuracy LogisticRegression: 78.15
Accuracy RandomForestClassifier: 76.47
Accuracy KNeighborsClassifier: 74.79
```

- com a combinação dos dois tratamentos

```
Accuracy LogisticRegression: 79.83
Accuracy RandomForestClassifier: 78.99
Accuracy KNeighborsClassifier: 78.15
```

*valores em torno de

6. Respostas às perguntas feitas nos itens 3 e 4

Pergunta feita no item 3:

• Qual modelo apresentou maior e menor acurácia? Indique possíveis justificativas para este tipo de comportamento, considerando o princípio de funcionamento dos algoritmos de aprendizagem supervisionada utilizados.

Maior: *Logistic Regression*

Menor: K-*Neighbors Classifier* (KNN)

6. Respostas às perguntas feitas nos itens 3 e 4

Perguntas feitas no item 4:

• O pré-processamento dos dados ajudou a melhorar a acurácia dos modelos? Justifique sua resposta.

Sim. Houve uma maior acurácia, porque limitou e selecionou mais os atributos, de forma que fosse possível um resultado mais satisfatório.

• Houve alguma diferença na acurácia quando duas melhorias foram adicionadas ao mesmo tempo e quando cada uma delas foi feita separadamente? Justifique sua resposta.

Houve uma melhora maior quando as duas técnica de tratamento foram adicionadas em conjunto. Cada pré-processamento colabora de formas diferentes para melhorar o desempenho.

7. Conclusões finais

A acurácia pode variar conforme as execuções, assim como varia de acordo a porcentagem de dados de treinamento e teste, que podem ser arbitradas conforme interesse do usuário.

O *LogisticRegression* e o *Random Forest Classifier* apresentaram um desempenho melhor que o KNN em mesmas condições e especificamente nesse tipo de dados tratados, o que não exclui a possibilidade desses algoritmos se comportarem de forma diferente quanto as acurácias, se aplicados a dados binários, por exemplo.

A escolha do algoritmo mais adequado depende da aplicação específica, de um modo geral todos os 3 apresentados entregam resultados satisfatórios para problemas o problema proposto.