PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA DE CIÊNCIAS EXATAS E DA COMPUTAÇÃO
CIÊNCIAS DA COMPUTAÇÃO

INTELIGÊNCIA ARTIFICIAL

CLARIMAR JOSE COELHO

ATIVIDADE EXTRA DISCIPLINAR III

MARCOS RODOLFO CRUVINEL GOULART QUERINO

GOIÂNIA
2020

INSTRUÇÕES

Escreva um programa em python para dados de credit_data.csv usando SVM para predizer se o cliente vai pagar ou não vai pagar o empréstimo.

ROTEIRO

1. Carga das bibliotecas
2. Pré-processamento:
    - Tratamento das idades inválidas (consistência de dados)
    - Dividir previsão de classe
    - Retire valores faltantes
    - Faça escalonamento das variáveis
3. Classificador:
    - Dividir a base de dados entre treinamento e teste
    - Construir o classificador com kernel rbf com custo de 2.0
    - Ajustar o modelo
    - Fazer a predição
    - Calcule a acurácia
    - Matriz de confusão

CÓDIGO

```
import pandas as pd
import numpy as np

# pre processamento
base= pd.read_csv('credit_data.csv')

# tratamento dos dados invalidos, por idade
# excluindo pessoas menores de 18, pois não podem arcar com a responsabilidade
# do exercicio proposto: pagamento de emprestimo
```

```
base.loc[base.age  < 18, 'age'] = 40.92

# divisao  de previsores  e classe
previsores=  base.iloc[:,  1:4].values
classe= base.iloc[:,  4].values

# uso do imputer  para retirar  retirar  os valores  faltantes
from sklearn.impute  import  SimpleImputer
imputer=  SimpleImputer(missing_values=  np.nan,  strategy=  'mean')
imputer=  imputer.fit(previsores[:,  1:4])
previsores[:,  1:4]= imputer.transform(previsores[:,  1:4])

# escalonamento  dos dados
from sklearn.preprocessing  import  StandardScaler
scaler= StandardScaler()
previsores=  scaler.fit_transform(previsores)

# dividindo  a base de dados em treinamento  e teste
from sklearn.model_selection  import  train_test_split
previsores_treinamento,  previsores_teste,  classe_treinamento,  classe_teste=
train_test_split(previsores,  classe,  test_size=  0.25, random_state=  0)

# importação  da biblioteca  SVM
from sklearn.svm  import  SVC

# construção  do classificador  com kernel rbf com custo  de 2.0
classificador=  SVC(kernel=  'rbf',  random_state=  1, C= 2.0)

# criação  do classificador
classificador.fit(previsores_treinamento,  classe_treinamento)

# pegar os 500 registros  de teste e submete  los  ao SVM para ver de qual parte
```

```
# da reta eles vao "cair", e gerar a respectiva classificação
previsoes= classificador.predict(previsores_teste)

# calculo de precisao e matriz de confusão
from sklearn.metrics import confusion_matrix, accuracy_score
precisao= accuracy_score(classe_teste, previsoes)
matriz= confusion_matrix(classe_teste, previsoes)
```

RESULTADOS

Figura 1 – explorador de variáveis

Name	Type	Size	Value
base	DataFrame	(2000, 5)	Column names: i#clientid, income, age, loan, c#default
classe	Array of int64	(2000,)	[0 0 0 ... 1 0 0]
classe_teste	Array of int64	(500,)	[1 0 0 ... 0 1 1]
classe_treinamento	Array of int64	(1500,)	[0 0 0 ... 0 0 0]
classificador	svm._classes.SVC	1	SVC object of sklearn.svm._classes module
imputer	impute._base.SimpleImputer	1	SimpleImputer object of sklearn.impute._base module
matriz	Array of int64	(2, 2)	[[434 2] [4 60]]
precisao	float64	1	0.988
previsoes	Array of int64	(500,)	[1 0 0 ... 0 1 1]
previsores	Array of float64	(2000, 3)	[[1.45393393 1.36538093 1.20281942] [-0.76217555 0.5426602 0.69 ...
previsores_teste	Array of float64	(500, 3)	[[1.59301567 -1.35435846 2.58262733] [0.99769755 0.99806572 0.84 ...
previsores_treinamento	Array of float64	(1500, 3)	[[-1.3754462 0.50631087 0.10980934] [1.45826409 -1.6489393 -1.21 ...
scaler	preprocessing._data.StandardScaler	1	StandardScaler object of sklearn.preprocessing._data module

Fonte: AED_III.py (em anexo) em python 3.8

Precisão = 0.988 = 98.8%.

Figura 2 – matriz de confusão

	0	1
0	434	2
1	4	60

Fonte: AED_III.py (em anexo) em python 3.8

BIBLIOGRAFIA

COUTINHO, Bernardo. Modelos de predição SVM. 2019. Disponível em: https://medium.com/turing-talks/turing-talks-12-classifica%C3%A7%C3%A3o-por-svm-f4598094a3f1. Acesso em 04 de novembro de 2020.

GRANATYR, Jones. **Machine Learning e Data Science com Python de A a Z.** Seção 9: Máquinas de vetores de suporte (SVM). 2020. Disponível em: https://www.udemy.com/course/machine-learning-e-data-science-com-python-y/. Acesso em 05 de novembro de 2020.

SANTANA, Rodrigo. **Classificando músicas do Spotify com SVM**. 2018. Disponível em: https://minerandodados.com.br/spotify-svm-python/. Acesso em 04 de novembro de 2020.

VAZ, Arthur Lamblet. **Classificando o paladar de receitas – SVM**. 2018. Disponível em: https://medium.com/@arthurlambletvaz/classificando-o-paladar-das-receitas-svm-bf0fbb185b10. Acesso em 04 de novembro de 2020.