Lista IA – 3

ALUNO: Caio Gomes Alcântara Glória

MATRICULA: 763989

PROFESSOR(A): Cristiane Neri

1-

- Total de registros = 14
- Jogou (Sim) = 9
- Não Jogou (Não) = 5

Questão 1

jogar		aparência			temperatura		umidade		ventando	
	sol	nublado	chuva	quente	agradável	frio	Alta	normal	sim	não
sim 9/14	2/9	4/9	3/9	2/9	4/9	3/9	3/9	6/9	3/9	6/9
não 5/14	3/5	0/5	2/5	2/5	2/5	1/5	4/5	1/5	3/5	3/5

P( )= probabilidade

P(jogar) = P(chuva) * P(fria) * P(normal) * P(ventando) * P(jogar) =(0,0158/soma) * 100 = 82,16%

P(~~jogar~~) = P(chuva) * P(fria) * P(normal) * P(ventando) * P(~~jogar~~) = (0,00343/SOMA) * 100 = 17,84%

2-

```python
# Importando bibliotecas necessárias
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split, RandomizedSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from imblearn.over_sampling import SMOTE

# Carregando a base de dados
df = pd.read_csv('titanic.csv')

# Visualizando os primeiros registros
print(df.head())

# Etapa 1: Pré-processamento dos dados
# Colunas 'Age', 'Fare' e 'Embarked' possuem valores ausentes
imputer = SimpleImputer(strategy='mean')
df['Age'] = imputer.fit_transform(df[['Age']])
df['Fare'] = imputer.fit_transform(df[['Fare']])
```

```
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])

# 1.2 Remover colunas desnecessárias ou redundantes
df = df.drop(columns=['Cabin', 'Ticket', 'Name', 'PassengerId'])

# 1.3 Codificação de variáveis categóricas
label_encoders = {}
for column in ['Sex', 'Embarked']:
    label_encoders[column] = LabelEncoder()
    df[column] = label_encoders[column].fit_transform(df[column])

# 1.4 Balanceamento dos dados com SMOTE
X = df.drop(columns='Survived')
y = df['Survived']

# Aplicando SMOTE para balancear
smote = SMOTE(random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)

# 1.5
X_train, X_test, y_train, y_test = train_test_split(X_resampled,
y_resampled, test_size=0.2, random_state=42)

# 1.6 Normalizando
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```