Campus Santo Amaro

Tecnologia em análise e desenvolvimento de sistemas

Logistics solution

São Paulo

2024

Colaboradores:

Erivan Santos Marques RA: 2223201673

```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

data = {
    'date': ['1.1.22', '2.1.22', '3.1.22', '4.1.22', '5.1.22'],
    'sales': [150, 200, 250, 300, 350],
    'customer': [120, 140, 160, 180, 200],
    'cost_of_operation': [5000, 5500, 6000, 6500, 7000],
    'profit_from_sales': [15000, 18000, 20000, 22000, 24500]
}

data_df = pd.DataFrame(data)

X = data_df[['customer', 'cost_of_operation', 'sales']]
y = data_df['profit_from_sales']

X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.3, random_state=42)

model = LinearRegression()
model.fit(X_train, y_train)

predictions = model.predict(X_val)

mae = mean_absolute_error(y_val, predictions)
mse = mean_squared_error(y_val, predictions)
rmse = mean_squared_error(y_val, predictions, squared=False)
r2 = r2_score(y_val, predictions)

print("Erro Médio Absoluto (MAE):", mae)
print("Erro Quadrático Médio (MSE):", mse)
print("Raiz do Erro Quadrático Médio (RMSE):", rmse)
print("R-quadrado (R²):", r2)

plt.figure(figsize=(10, 6))
plt.scatter(y_val, predictions, color='blue', alpha=0.5)
plt.plot([y_val.min(), y_val.max()], [y_val.min(), y_val.max()], 'k--', lw=2)
plt.xlabel('Valores Reais')
plt.ylabel('Previsões')
plt.title('Valores Reais vs. Previsões')
plt.show()


data_df.to_csv("logistics_solution_data.csv", index=False)

print("Arquivo 'logistics_solution_data.csv' salvo com sucesso!")
```

Aprendizado de Máquina (Algoritmos de ML)

* Entrega 1: Exploração de Dados e Pré-processamento:

```
Comece a programar ou gere código com IA.


import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

data = {
    'date': ['1.1.22', '2.1.22', '3.1.22', '4.1.22', '5.1.22'],
    'sales': [150, 200, 250, 300, 350],
    'customer': [120, 140, 160, 180, 200],
    'cost_of_operation': [5000, 5500, 6000, 6500, 7000],
    'profit_from_sales': [15000, 18000, 20000, 22000, 24500]
}

df = pd.DataFrame(data)

X = df.drop(["profit_from_sales", "date"], axis=1)
y = df["profit_from_sales"]
y = pd.cut(y, bins=2, labels=['baixo venda', 'alta venda'])

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train, X_val, y_train, y_val = train_test_split(X_scaled, y, test_size=0.3, random_state=42)


knn_cls = KNeighborsClassifier(n_neighbors=5)
knn_cls.fit(X_train, y_train)

y_pred = knn_cls.predict(X_val)

accuracy = accuracy_score(y_val, y_pred)
precision = precision_score(y_val, y_pred, average='weighted', zero_division='warn')
recall = recall_score(y_val, y_pred, average='weighted', zero_division='warn')
f1 = f1_score(y_val, y_pred, average='weighted', zero_division='warn')

print("Acurácia:", accuracy)
print("Precisão:", precision)
print("Recall:", recall)
print("F1-score:", f1)


cn = confusion_matrix(y_val, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cn, annot=True, fmt="d", cmap="Blues", cbar=False)
plt.xlabel('Predicted labels')
plt.ylabel('True labels')
plt.title('Confusion Matrix')
plt.show()
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-1-84209fd3fce1> in <cell line: 37>()
     35 knn_cls.fit(X_train, y_train)
     36
---> 37 y_pred = knn_cls.predict(X_val)
     38
     39 accuracy = accuracy_score(y_val, y_pred)

                              1 frames
/usr/local/lib/python3.10/dist-packages/sklearn/neighbors/_base.py in kneighbors(self, X, n_neighbors, return_distance)
    808         n_samples_fit = self.n_samples_fit_
    809         if n_neighbors > n_samples_fit:
--> 810             raise ValueError(
    811                 "Expected n_neighbors <= n_samples, "
    812                 " but n_samples = %d, n_neighbors = %d" % (n_samples_fit, n_neighbors)

ValueError: Expected n_neighbors <= n_samples,  but n_samples = 3, n_neighbors = 5
```

* Entrega 2: Implementação de Modelos de Aprendizado de Máquina: