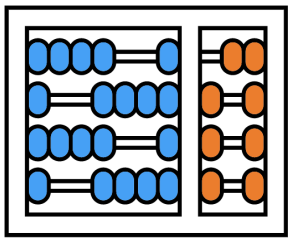


Instituto de Computação (IC) Unicamp

Escola de Extensão da Unicamp

Universidade Estadual de Campinas

INF 331 - Componentização e Reuso de Software
Prof. André Santanché

Aula 03

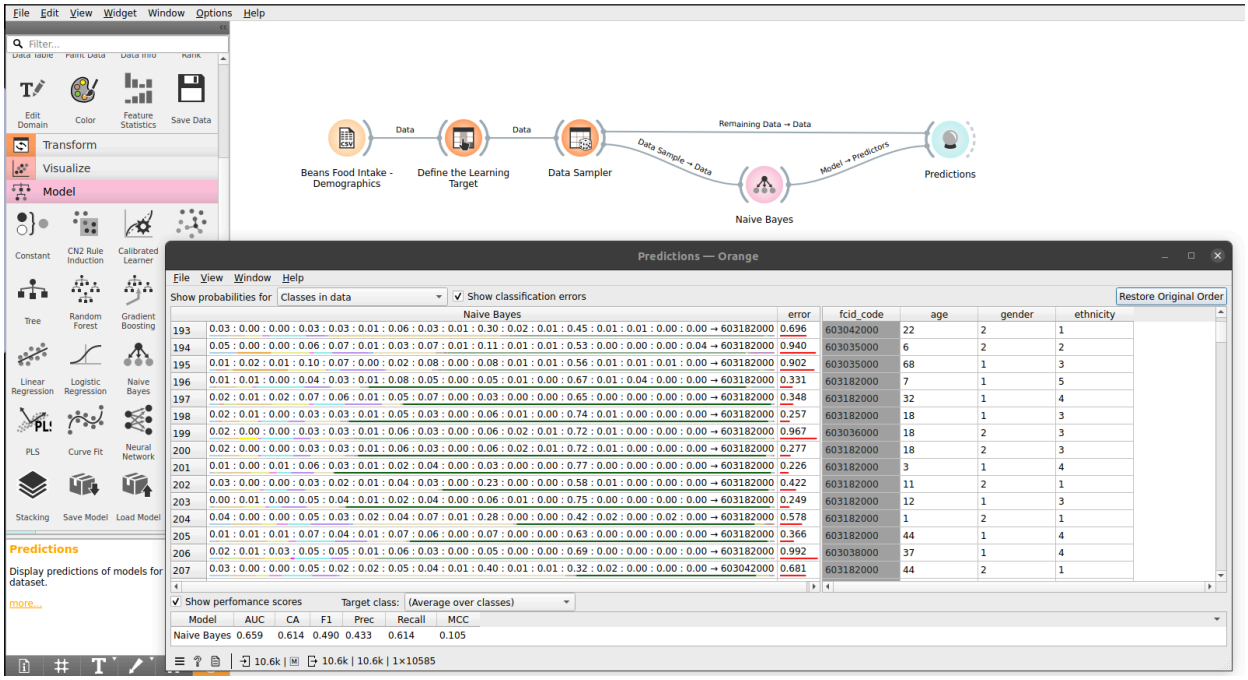
componentização e serviços web

Aluno

Douglas Sermarini - 147730

Repositório : https://github.com/Douglas019BR/INF331-lab03

Laboratório 1 – Tarefa 1
Arquivo: aula3-lab1-task1.ows



Laboratório 2 – Tarefa 1
Arquivo: aula3-lab2-task1.ipynb

[8]:
```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import accuracy_score

class FoodEthnicityPredictor:
    def __init__(self):
        self.model = LogisticRegression()
        self.le_gender = LabelEncoder()
        self.le_ethnicity = LabelEncoder()
        self.le_fcid = LabelEncoder()

    def train(self, csv_file_path):
        # Carrega os dados
        data = pd.read_csv(csv_file_path)

        # Codifica as variáveis categóricas
        data['gender'] = self.le_gender.fit_transform(data['gender'])
        data['ethnicity'] = self.le_ethnicity.fit_transform(data['ethnicity'])
        data['fcid_code'] = self.le_fcid.fit_transform(data['fcid_code'])

        # Separa as características e o alvo
        X = data[['age', 'gender', 'fcid_code']]
        y = data['ethnicity']

        # Divide os dados em conjuntos de treino e teste
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
        # Treina o modelo
        self.model.fit(X_train, y_train)

        # Avalia o modelo
        y_pred = self.model.predict(X_test)
        accuracy = accuracy_score(y_test, y_pred)
        print(f"Acurácia do modelo: {accuracy:.2f}")

    def predict(self, age, gender, fcid_code):
        # Codifica os dados de entrada
        gender_encoded = self.le_gender.transform([gender])[0]
        fcid_encoded = self.le_fcid.transform([fcid_code])[0]

        # Faz a predição
        prediction = self.model.predict([[age, gender_encoded, fcid_encoded]])

        # Decodifica a predição
        ethnicity = self.le_ethnicity.inverse_transform(prediction)[0]
        return ethnicity
```

[11]:
```
predictor = FoodEthnicityPredictor()
predictor.train("intake-person-demo(beans).csv")
predicted_ethnicity = predictor.predict(30, 1, '603182000')
print(f"Etnia prevista: {predicted_ethnicity}")
```

```
Acurácia do modelo: 0.40
Etnia prevista: 3
/srv/conda/envs/notebook/lib/python3.10/site-packages/sklearn/linear_model/_logistic.py:469: ConvergenceWarning: lbfgs failed to converge (status=
1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  n_iter_i = _check_optimize_result(
/srv/conda/envs/notebook/lib/python3.10/site-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but LogisticRegressio
n was fitted with feature names
  warnings.warn(
```