```
# Customer Purchase Prediction using Decision Tree
# Complete Combined Implementation

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.tree import DecisionTreeClassifier, plot_tree
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, f1_score


data = pd.read_csv("customer_data.csv")  # Replace with your file name

# Target column (0 = No Purchase, 1 = Purchase)
X = data.drop("purchase", axis=1)
y = data["purchase"]



numeric_features = ["age", "balance"]
categorical_features = ["job", "marital", "education", "contact"]
```

```
numeric_transformer = Pipeline(steps=[
    ("imputer", SimpleImputer(strategy="median"))
])

categorical_transformer = Pipeline(steps=[
    ("imputer", SimpleImputer(strategy="most_frequent")),
    ("onehot", OneHotEncoder(handle_unknown="ignore"))
])

preprocessor = ColumnTransformer(
    transformers=[
        ("num", numeric_transformer, numeric_features),
        ("cat", categorical_transformer, categorical_features)
    ]
)


pipeline = Pipeline(steps=[
    ("preprocessor", preprocessor),
    ("classifier", DecisionTreeClassifier(random_state=42))
])


X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)
```

```
param_grid = {
    "classifier__max_depth": [3, 5, 7, 10],
    "classifier__min_samples_split": [2, 5, 10],
    "classifier__criterion": ["gini", "entropy"]
}

grid_search = GridSearchCV(
    pipeline,
    param_grid,
    cv=5,
    scoring="f1",
    n_jobs=-1
)

grid_search.fit(X_train, y_train)

best_model = grid_search.best_estimator_

print("Best Parameters:", grid_search.best_params_)



y_pred = best_model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print("\nModel Accuracy:", round(accuracy * 100, 2), "%")
print("F1 Score:", round(f1, 4))
print("\nClassification Report:\n")
print(classification_report(y_test, y_pred))
```

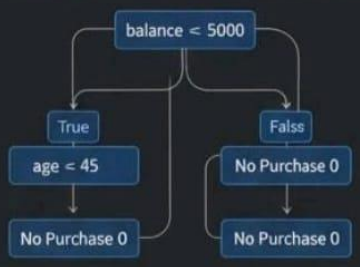
```
cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(6,5))
sns.heatmap(cm, annot=True, fmt="d", cmap="Blues",
        xticklabels=["No Purchase", "Purchase"],
        yticklabels=["No Purchase", "Purchase"])
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()



# Get feature names after one-hot encoding
ohe = best_model.named_steps["preprocessor"]\
            .named_transformers_["cat"]\
            .named_steps["onehot"]

encoded_features = list(ohe.get_feature_names_out(categorical_features))
all_features = numeric_features + encoded_features

importances = best_model.named_steps["classifier"].feature_importances_

feature_importance_df = pd.DataFrame({
   "Feature": all_features,
   "Importance": importances
}).sort_values(by="Importance", ascending=False)

print("\nTop Important Features:\n")
print(feature_importance_df.head(10))
```

```
# Plot Feature Importance
plt.figure(figsize=(10,6))
sns.barplot(x="Importance", y="Feature",
        data=feature_importance_df.head(10))
plt.title("Top 10 Important Features")
plt.show()



plt.figure(figsize=(20,10))
plot_tree(
   best_model.named_steps["classifier"],
   filled=True,
   fontsize=8
)
plt.title("Decision Tree Visualization")
plt.show()



with open("customer_purchase_model.pkl", "wb") as file:
   pickle.dump(best_model, file)

print("\nModel saved as customer_purchase_model.pkl")
```

1. Data Dictionary & System Encoding

Variable	Logic Role	System Constraint
age	Continouus	≥ 18
job	One-Hot Encoding	✓
martial	3 Sparse Columns	✓
Ordinal	Float (Euros)	✓
education	Float '(Euros) 0	✓
balance	Impute "Unknown"	✓
contact	Mode-fill strategy	✓
contact		

2. Logical Branching (Visual Schema)



3. Core Implementation Code

```
import pandas  ts pd pd
21 < // = 1ursact 0aes <>
21 ir nport = vaen((art.mangatien'ineun");
23 {
23   import = fmaen slienns" Decision slarfper));
20   {
25     = DT_CONFIGG = (pd
24     from  skleean.tree.tree DecisionClackClcrforgier, plot_tree
26     DT_CONFIGT =;
26
27       dT_pltoct = fontact = "manattllaien(.s)); )));
28   )-
27     );
38 );
36 ogoor;
35 {
35   int = dery.maoris.laeclsion"$inanggiet, plot_sqd() ));
35     = = Beu lctaube;
```

Model Accuracy