

Talal Kamal

Project 2-2: Titanic Dataset Machine Learning

This report includes dataset preprocessing, machine learning models with Grid Search, feature selection methods, and evaluation results based on the Titanic dataset.

Python Code:

```
# Project 2-2: Titanic Dataset Machine Learning

import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.metrics import accuracy_score, f1_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.feature_selection import SelectKBest, mutual_info_classif, RFE

df = pd.read_csv("Titanic-Dataset.csv")

# Step 1 & 2: Cleaning
X = df.drop(columns=['Survived', 'Cabin', 'Name', 'Ticket', 'PassengerId'])
y = df['Survived']

numeric_features = ['Age', 'Fare', 'SibSp', 'Parch', 'Pclass']
categorical_features = ['Sex', 'Embarked']

preprocessor = ColumnTransformer([
    ('num', Pipeline([
        ('imputer', SimpleImputer(strategy='median')),
        ('scaler', StandardScaler())
    ]), numeric_features),
    ('cat', Pipeline([
        ('imputer', SimpleImputer(strategy='most_frequent')),
        ('onehot', OneHotEncoder(handle_unknown='ignore'))
    ]), categorical_features)
])

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

# Step 3: Grid Search Models
pipe_lr = Pipeline([('prep', preprocessor), ('model', LogisticRegression(max_iter=1000))])
param_lr = {'model__C':[0.1,1,10], 'model__solver':['liblinear','lbfgs']}
gs_lr = GridSearchCV(pipe_lr, param_lr, cv=5, scoring='accuracy')
gs_lr.fit(X_train, y_train)

pipe_rf = Pipeline([('prep', preprocessor), ('model', RandomForestClassifier(random_state=42))])
param_rf = {'model__n_estimators':[100,200], 'model__max_depth':[4,6,None]}
gs_rf = GridSearchCV(pipe_rf, param_rf, cv=5, scoring='accuracy')
gs_rf.fit(X_train, y_train)

# Step 4: Feature Selection
prep_no_scale = ColumnTransformer([
    ('num', Pipeline([('imputer', SimpleImputer(strategy='median'))]), numeric_features),
    ('cat', Pipeline([
        ('imputer', SimpleImputer(strategy='most_frequent')),
        ('onehot', OneHotEncoder(handle_unknown='ignore'))
    ]), categorical_features)
])

X_train_t = prep_no_scale.fit_transform(X_train)
X_test_t = prep_no_scale.transform(X_test)

skb = SelectKBest(score_func=mutual_info_classif, k=5)
X_train_skb = skb.fit_transform(X_train_t, y_train)

rfe = RFE(LogisticRegression(max_iter=1000, solver='liblinear'), n_features_to_select=5)
X_train_rfe = rfe.fit_transform(X_train_t, y_train)
print("Done")
```