```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
from skopt import BayesSearchCV
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import TomekLinks, RandomUnderSampler
from sklearn.impute import SimpleImputer, KNNImputer

# Carregar os dados
df_train = pd.read_csv(r'Lista 2\train.csv')
df_test = pd.read_csv(r'Lista 2\test.csv')

def preprocess_data(df):
    df = df.drop(['Name', 'Ticket', 'Cabin'], axis=1, errors='ignore')
# Remover colunas irrelevantes
    df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})  #
Transformação categórica
    df = pd.get_dummies(df, columns=['Embarked'], drop_first=True)  #
One-hot encoding
    return df

# Processar dados
df_train = preprocess_data(df_train)
X = df_train.drop(columns=['Survived'])
y = df_train['Survived']

# Imputação de dados ausentes
imputer = KNNImputer(n_neighbors=5)
X = pd.DataFrame(imputer.fit_transform(X), columns=X.columns)

# Dividir em treino e validação
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2,
random_state=42)

# Balanceamento usando SMOTE
smote = SMOTE(random_state=42)
X_train_sm, y_train_sm = smote.fit_resample(X_train, y_train)

# Ajuste de hiperparâmetros com BayesSearchCV
```

```
rf = RandomForestClassifier(random_state=42)
param_grid_rf = {'n_estimators': (50, 500), 'max_depth': (3, 15),
'min_samples_split': (2, 10)}
bayes_search_rf = BayesSearchCV(rf, param_grid_rf, n_iter=20, cv=3,
random_state=42)
bayes_search_rf.fit(X_train_sm, y_train_sm)

# Modelo final
y_pred_rf = bayes_search_rf.best_estimator_.predict(X_val)

# Avaliação
metrics_rf = {
    'Accuracy': accuracy_score(y_val, y_pred_rf),
    'Precision': precision_score(y_val, y_pred_rf),
    'Recall': recall_score(y_val, y_pred_rf),
    'F1-Score': f1_score(y_val, y_pred_rf)
}

# Exibir resultados
print("Melhores hiperparâmetros:", bayes_search_rf.best_params_)
print("Métricas Random Forest:", metrics_rf)
```

Melhores hiperparâmetros:
```
OrderedDict({'max_depth': 15, 'min_samples_split': 2, 'n_estimators': 446})
```

Métricas Random Forest:
```
{'Accuracy': 0.7932960893854749, 'Precision': 0.7846153846153846, 'Recall':
0.6891891891891891, 'F1-Score': 0.7338129496402878}
```