

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.impute import SimpleImputer

from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

from sklearn.feature_selection import SelectKBest, chi2, RFE
from sklearn.metrics import accuracy_score, classification_report
```

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

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	
2	3	1	3	Heikinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	

Next steps: [Generate code with df](#) [New interactive sheet](#)

```
# Drop useless columns
df.drop(['Cabin', 'Name', 'Ticket'], axis=1, inplace=True)

# Fill missing values
df['Age'] = df['Age'].fillna(df['Age'].mean())
df['Embarked'] = df['Embarked'].fillna(df['Embarked'].mode()[0])
```

```
le = LabelEncoder()

df['Sex'] = le.fit_transform(df['Sex'])
df['Embarked'] = le.fit_transform(df['Embarked'])
```

```
X = df.drop('Survived', axis=1)
y = df['Survived']
```

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

```
rf = RandomForestClassifier()

param_grid = {
    'n_estimators': [50, 100],
    'max_depth': [3, 5, 10]
}

grid_rf = GridSearchCV(rf, param_grid, cv=5)
grid_rf.fit(X_train, y_train)

best_rf = grid_rf.best_estimator_

y_pred_rf = best_rf.predict(X_test)

print("Random Forest Accuracy:", accuracy_score(y_test, y_pred_rf))
```

Random Forest Accuracy: 0.8100558659217877

```
lr = LogisticRegression(max_iter=1000)

param_grid = {
    'C': [0.1, 1, 10]
}

grid_lr = GridSearchCV(lr, param_grid, cv=5)
grid_lr.fit(X_train, y_train)
```

```
best_lr = grid_lr.best_estimator_  
  
y_pred_lr = best_lr.predict(X_test)  
  
print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred_lr))
```

Logistic Regression Accuracy: 0.8044692737430168

```
selector = SelectKBest(score_func=chi2, k=5)  
X_new = selector.fit_transform(X, y)  
  
selected_features = X.columns[selector.get_support()]  
print("Selected Features (Chi2):", selected_features)
```

Selected Features (Chi2): Index(['Pclass', 'Sex', 'Age', 'Fare', 'Embarked'], dtype='object')

```
model = RandomForestClassifier()  
rfe = RFE(model, n_features_to_select=5)  
  
X_rfe = rfe.fit_transform(X, y)  
  
selected_features_rfe = X.columns[rfe.support_]  
print("Selected Features (RFE):", selected_features_rfe)
```

Selected Features (RFE): Index(['PassengerId', 'Pclass', 'Sex', 'Age', 'Fare'], dtype='object')

```
X_selected = df[selected_features]  
  
X_train, X_test, y_train, y_test = train_test_split(  
    X_selected, y, test_size=0.2, random_state=42  
)  
  
model = RandomForestClassifier()  
model.fit(X_train, y_train)  
  
y_pred = model.predict(X_test)  
  
print("Accuracy after Feature Selection:", accuracy_score(y_test, y_pred))
```

Accuracy after Feature Selection: 0.7877094972067039