

dataset <https://www.kaggle.com/competitions/spaceship-titanic/data>

dataset <https://www.kaggle.com/competitions/spaceship-titanic/data>

```
pip install pandas scikit-learn
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (2.2.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.12/dist-packages (1.6.1)
Requirement already satisfied: numpy>=1.26.0 in /usr/local/lib/python3.12/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas) (2025.3)
Requirement already satisfied: scipy>=1.6.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.16.3)
Requirement already satisfied: joblib>=1.2.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (1.5.3)
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn) (3.6.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
```

```
import pandas as pd
from sklearn.model_selection import GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.impute import SimpleImputer
from sklearn.pipeline import Pipeline
from sklearn.feature_selection import SelectKBest, chi2
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, accuracy_score
```

```
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")

y = train["Survived"]

X = train.drop(["Survived", "Name", "Ticket", "Cabin"], axis=1)
test = test.drop(["Name", "Ticket", "Cabin"], axis=1)

cat_cols = ["Sex", "Embarked"]
X = pd.get_dummies(X, columns=cat_cols)
test = pd.get_dummies(test, columns=cat_cols)

for c in X.columns:
    if c not in test.columns:
        test[c] = 0
```

```
pipeline = Pipeline([
    ("imputer", SimpleImputer(strategy="median")),
    ("scaler", StandardScaler())
])

X_processed = pipeline.fit_transform(X)
test_processed = pipeline.transform(test)
```

```
selector = SelectKBest(score_func=chi2, k=10)
X_selected = selector.fit_transform(abs(X_processed), y)
test_selected = selector.transform(abs(test_processed))
```

```
models = {
    "LogisticRegression": LogisticRegression(max_iter=500),
    "RandomForest": RandomForestClassifier()
}

params = {
    "LogisticRegression": {"C": [0.1, 1, 10]},
    "RandomForest": {"n_estimators": [50, 100, 200], "max_depth": [5, 10]}
}

for name in models:
    grid = GridSearchCV(models[name], params[name], cv=5)
    grid.fit(X_selected, y)
    pred = grid.predict(X_selected)
    print(name)
```

```
print(accuracy_score(y, pred))
print(classification_report(y, pred))
```

```
LogisticRegression
0.7991021324354658
```

	precision	recall	f1-score	support
0	0.81	0.88	0.84	549
1	0.77	0.67	0.72	342
accuracy			0.80	891
macro avg	0.79	0.78	0.78	891
weighted avg	0.80	0.80	0.80	891

```
RandomForest
0.9438832772166106
```

	precision	recall	f1-score	support
0	0.93	0.98	0.96	549
1	0.97	0.88	0.92	342
accuracy			0.94	891
macro avg	0.95	0.93	0.94	891
weighted avg	0.95	0.94	0.94	891

```
final_model = GridSearchCV(RandomForestClassifier(), params["RandomForest"], cv=5)
final_model.fit(X_selected, y)

submission = final_model.predict(test_selected)

output = pd.DataFrame({
    "PassengerId": test["PassengerId"],
    "Survived": submission
})

output.to_csv("submission.csv", index=False)
```