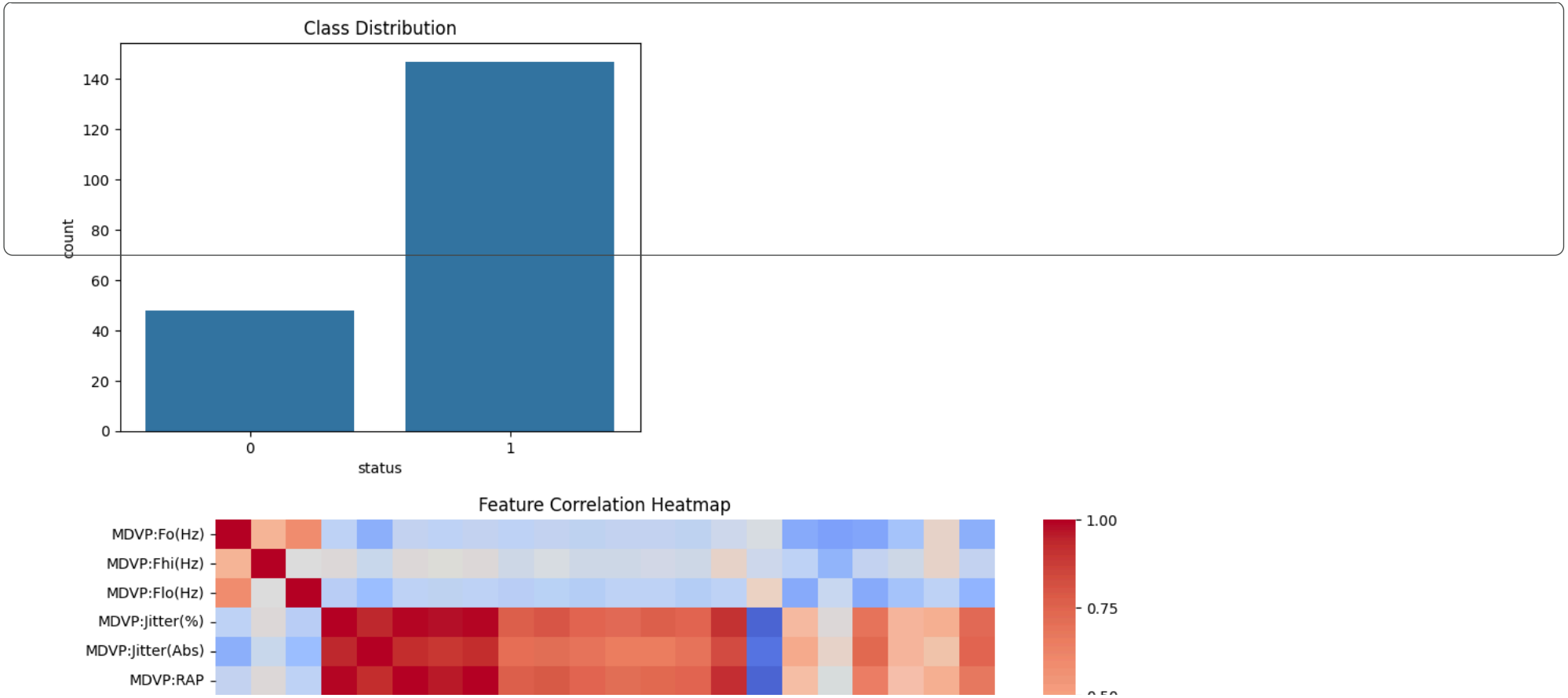
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler
from sklearn.utils.class_weight import compute_class_weight
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

from xgboost import XGBClassifier, plot_importance

df = pd.read_csv("parkinsons.data")
df.shape
df.head()

df.drop(columns=['name'], inplace=True)

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

sns.countplot(x=y)
plt.title("Class Distribution")
plt.show()

class_weights = compute_class_weight(
    class_weight='balanced',
    classes=np.unique(y),
    y=y
)

scale_pos_weight = class_weights[0] / class_weights[1]
scale_pos_weight

plt.figure(figsize=(12,8))
sns.heatmap(X.corr(), cmap='coolwarm')
plt.title("Feature Correlation Heatmap")
plt.show()

corr_matrix = X.corr().abs()
upper_triangle = corr_matrix.where(
    np.triu(np.ones(corr_matrix.shape), k=1).astype(bool)
)
drop_features = [
    column for column in upper_triangle.columns
    if any(upper_triangle[column] > 0.9)
]
X = X.drop(columns=drop_features)
drop_features

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

scaler = StandardScaler()
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

xgb_baseline = XGBClassifier(
    n_estimators=100,
    max_depth=4,
    learning_rate=0.1,
    subsample=0.8,
    colsample_bytree=0.8,
    scale_pos_weight=scale_pos_weight,
    eval_metric='logloss',
    random_state=42
)

xgb_baseline.fit(X_train, y_train)

baseline_pred = xgb_baseline.predict(X_test)
accuracy_score(y_test, baseline_pred)
classification_report(y_test, baseline_pred)

param_grid = {
    'n_estimators': [100, 200],
    'max_depth': [3, 4, 5],
    'learning_rate': [0.01, 0.1],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}

xgb_model = XGBClassifier(
    scale_pos_weight=scale_pos_weight,
    eval_metric='logloss',
    random_state=42
)

grid_search = GridSearchCV(
    estimator=xgb_model,
    param_grid=param_grid,
    scoring='f1',
    cv=5,
    n_jobs=-1,
    verbose=1
)

grid_search.fit(X_train, y_train)

grid_search.best_params_

best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)

accuracy_score(y_test, y_pred)
classification_report(y_test, y_pred)

cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
```

```
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()

plt.figure(figsize=(10,6))
plot_importance(best_model, max_num_features=10)
plt.title("Feature Importance")
plt.show()
```


Class Distribution
count
140
120
100
80
60
40
20
0
0
1
status
Feature Correlation Heatmap
MDVP:Fo(Hz)
MDVP:Fhi(Hz)
MDVP:Flo(Hz)
MDVP:Jitter(%)
MDVP:Jitter(Abs)
MDVP:RAP
1.00
0.75