```
import pandas as pd
import numpy as np
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import KFold, cross_val_score
from sklearn.preprocessing import scale
from sklearn.metrics import accuracy_score

# 1. Загрузка данных из файла wine.data
# Указываем, что файл без заголовков и используем названия колонок из описания
column_names = [
    'Class',
    'Alcohol',
    'Malic acid',
    'Ash',
    'Alcalinity of ash',
    'Magnesium',
    'Total phenols',
    'Flavanoids',
    'Nonflavanoid phenols',
    'Proanthocyanins',
    'Color intensity',
    'Hue',
    'OD280/OD315 of diluted wines',
    'Proline'
]
data = pd.read_csv('wine.data', header=None, names=column_names)

X = data.drop('Class', axis=1)
y = data['Class']

print(f"Размерность данных: {X.shape}")
print(f"Количество классов: {len(np.unique(y))}")
print(f"Количество признаков: {X.shape[1]}")

# Создание генератора разбиений для кросс-валидации
kf = KFold(n_splits=5, shuffle=True, random_state=42)

# Поиск оптимального k без масштабирования
print("\n--- Поиск оптимального k БЕЗ масштабирования ---")
accuracies = []
best_k1 = 1
best_accuracy1 = 0
for k in range(1, 51):
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X, y, cv=kf, scoring='accuracy')
    mean_accuracy = np.mean(scores)
    accuracies.append(mean_accuracy)
    if mean_accuracy > best_accuracy1:
        best_accuracy1 = mean_accuracy
        best_k1 = k
print(f"Оптимальное k без масштабирования: {best_k1}")
print(f"Точность при k={best_k1}: {best_accuracy1:.2f}")

# Масштабирование признаков и повторный поиск
print("\n--- Поиск оптимального k С масштабированием ---")
X_scaled = scale(X)
accuracies_scaled = []
best_k2 = 1
best_accuracy2 = 0
for k in range(1, 51):
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X_scaled, y, cv=kf, scoring='accuracy')
    mean_accuracy = np.mean(scores)
    accuracies_scaled.append(mean_accuracy)
    if mean_accuracy > best_accuracy2:
        best_accuracy2 = mean_accuracy
        best_k2 = k
print(f"Оптимальное k с масштабированием: {best_k2}")
print(f"Точность при k={best_k2}: {best_accuracy2:.2f}")

# Сравнение результатов
print("\n--- Сравнение результатов ---")
print(f"Улучшение точности: {best_accuracy2 - best_accuracy1:.2f}")
print(f"Масштабирование признаков помогло: {'ДА' if best_accuracy2 > best_accuracy1 else 'НЕТ'}")

# Ответы на вопросы
print("\n=== ОТВЕТЫ НА ВОПРОСЫ ===")
print(f"1. Оптимальное k без масштабирования: {best_k1}")
print(f"2. Точность при этом k: {best_accuracy1:.2f}")
print(f"3. Оптимальное k с масштабированием: {best_k2}")
print(f"4. Точность при этом k: {best_accuracy2:.2f}")
```

```
print(f"5. Масштабирование признаков помогло: ДА")
```

```
Размерность данных: (178, 13)
Количество классов: 3
Количество признаков: 13

--- Поиск оптимального k БЕЗ масштабирования ---
Оптимальное k без масштабирования: 1
Точность при k=1: 0.73

--- Поиск оптимального k С масштабированием ---
Оптимальное k с масштабированием: 29
Точность при k=29: 0.98

--- Сравнение результатов ---
Улучшение точности: 0.25
Масштабирование признаков помогло: ДА

=== ОТВЕТЫ НА ВОПРОСЫ ===
1. Оптимальное k без масштабирования: 1
2. Точность при этом k: 0.73
3. Оптимальное k с масштабированием: 29
4. Точность при этом k: 0.98
5. Масштабирование признаков помогло: ДА
```

Напишите программный код или сгенерируйте его с помощью искусственного интеллекта.