```python
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score

import torch
import torch.nn as nn
import torch.optim as optim
from torch.utils.data import TensorDataset, DataLoader
import kagglehub


path = kagglehub.dataset_download("sukhmandeepsinghbrar/housing-price-
dataset")
print("Path to dataset files:", path)

housing_data = pd.read_csv(f"{path}/Housing.csv")

housing_data = pd.get_dummies(housing_data, drop_first=True)

X = housing_data.drop("price",
axis=1).select_dtypes(include=[np.number]).values
y = housing_data["price"].values.reshape(-1, 1)
```

```
Path to dataset files:
/root/.cache/kagglehub/datasets/sukhmandeepsinghbrar/housing-price-
dataset/versions/1
```

```python
X = torch.tensor(X, dtype=torch.float32)
y = torch.tensor(y, dtype=torch.float32)

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

scaler = StandardScaler()
X_train_scaled = torch.tensor(scaler.fit_transform(X_train.numpy()),
dtype=torch.float32)
X_test_scaled = torch.tensor(scaler.transform(X_test.numpy()),
dtype=torch.float32)

train_dataset = TensorDataset(X_train_scaled, y_train)
test_dataset = TensorDataset(X_test_scaled, y_test)
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)

class MLPModel(nn.Module):
    def __init__(self):
        super().__init__()
```

```
        self.layers = nn.Sequential(
            nn.Linear(X_train.shape[1], 64),
            nn.ReLU(),
            nn.Linear(64, 32),
            nn.ReLU(),
            nn.Linear(32, 1)
        )

    def forward(self, x):
        return self.layers(x)


device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = MLPModel().to(device)
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)

def train_model(model, train_loader, criterion, optimizer,
epochs=200):
    train_losses = []

    for epoch in range(epochs):
        model.train()
        running_loss = 0.0

        for inputs, targets in train_loader:
            inputs, targets = inputs.to(device), targets.to(device)

            optimizer.zero_grad()
            outputs = model(inputs)
            loss = criterion(outputs, targets)
            loss.backward()
            optimizer.step()

            running_loss += loss.item() * inputs.size(0)

        epoch_loss = running_loss / len(train_loader.dataset)
        train_losses.append(epoch_loss)
        print(f"Epoch {epoch+1}/{epochs}, Loss: {epoch_loss:.4f}")

    return train_losses

train_losses = train_model(model, train_loader, criterion, optimizer,
epochs=200)

plt.plot(range(1, 201), train_losses, label='Train Loss')
plt.xlabel('Epochs')
plt.ylabel('Mean Squared Error')
plt.title('Training Loss over Epochs')
plt.legend()
plt.show()
```

```python
def evaluate_model(model, test_loader, criterion):
    model.eval()
    test_loss = 0.0
    y_true = []
    y_pred = []

    with torch.no_grad():
        for inputs, targets in test_loader:
            inputs, targets = inputs.to(device), targets.to(device)
            outputs = model(inputs)
            loss = criterion(outputs, targets)
            test_loss += loss.item() * inputs.size(0)
            y_true.extend(targets.cpu().numpy())
            y_pred.extend(outputs.cpu().numpy())

    test_loss /= len(test_loader.dataset)
    return test_loss, y_true, y_pred
```

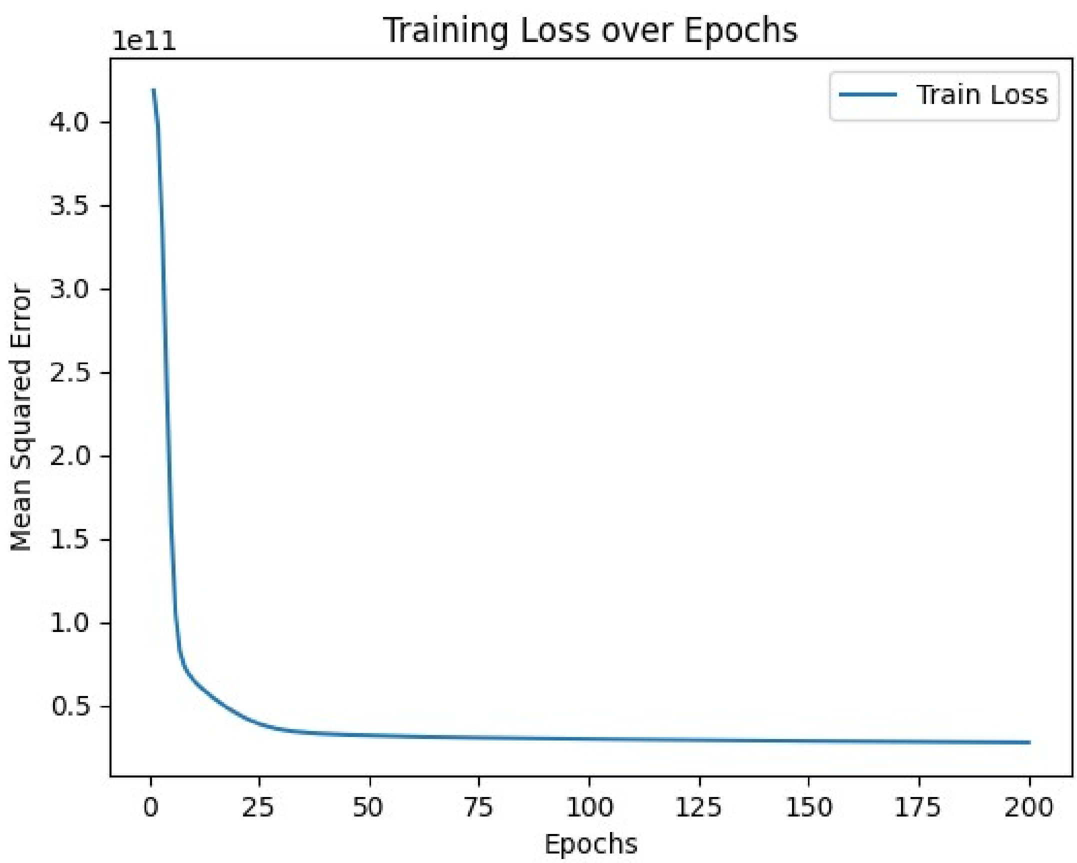
```
Epoch 1/200, Loss: 418330446235.9690
Epoch 2/200, Loss: 396046121554.3820
Epoch 3/200, Loss: 334024750750.7822
Epoch 4/200, Loss: 242546482765.0517
Epoch 5/200, Loss: 157069545002.4643
Epoch 6/200, Loss: 104701183200.9365
Epoch 7/200, Loss: 82437728109.1220
Epoch 8/200, Loss: 73262465436.2059
Epoch 9/200, Loss: 68387595632.0241
Epoch 10/200, Loss: 65068174583.4420
Epoch 11/200, Loss: 62344190325.1174
Epoch 12/200, Loss: 59912845661.9012
Epoch 13/200, Loss: 57649130174.8821
Epoch 14/200, Loss: 55526006903.3976
Epoch 15/200, Loss: 53490772311.7418
Epoch 16/200, Loss: 51561035704.2193
Epoch 17/200, Loss: 49734167379.6553
Epoch 18/200, Loss: 47992191618.1173
Epoch 19/200, Loss: 46340818905.3853
Epoch 20/200, Loss: 44802353942.0613
Epoch 21/200, Loss: 43348820971.0344
Epoch 22/200, Loss: 42016951023.2097
Epoch 23/200, Loss: 40793402333.0573
Epoch 24/200, Loss: 39692190887.6067
Epoch 25/200, Loss: 38699529011.4369
Epoch 26/200, Loss: 37806336581.8263
Epoch 27/200, Loss: 37028505797.3377
Epoch 28/200, Loss: 36349316646.7331
Epoch 29/200, Loss: 35790972919.2347
Epoch 30/200, Loss: 35266192021.8984
Epoch 31/200, Loss: 34841650664.7838
```

```
Epoch 32/200, Loss: 34473356403.2518
Epoch 33/200, Loss: 34153665342.0974
Epoch 34/200, Loss: 33875309405.1313
Epoch 35/200, Loss: 33634649482.3200
Epoch 36/200, Loss: 33444757126.3815
Epoch 37/200, Loss: 33235167750.1002
Epoch 38/200, Loss: 33069923956.3771
Epoch 39/200, Loss: 32915764462.9136
Epoch 40/200, Loss: 32782182949.8448
Epoch 41/200, Loss: 32648217727.0968
Epoch 42/200, Loss: 32531911118.7248
Epoch 43/200, Loss: 32412756256.8995
Epoch 44/200, Loss: 32316389228.5890
Epoch 45/200, Loss: 32197671855.5725
Epoch 46/200, Loss: 32121107112.3766
Epoch 47/200, Loss: 32001397602.1062
Epoch 48/200, Loss: 31929124154.2478
Epoch 49/200, Loss: 31851157360.3202
Epoch 50/200, Loss: 31771855548.9869
Epoch 51/200, Loss: 31710061222.1557
Epoch 52/200, Loss: 31633686811.0954
Epoch 53/200, Loss: 31554652651.8043
Epoch 54/200, Loss: 31500711763.4184
Epoch 55/200, Loss: 31439792465.5824
Epoch 56/200, Loss: 31370355160.4377
Epoch 57/200, Loss: 31304165392.5830
Epoch 58/200, Loss: 31251320112.2980
Epoch 59/200, Loss: 31183926823.8584
Epoch 60/200, Loss: 31144965141.9132
Epoch 61/200, Loss: 31075227021.8735
Epoch 62/200, Loss: 31024575329.6324
Epoch 63/200, Loss: 30968871825.0124
Epoch 64/200, Loss: 30920882932.3031
Epoch 65/200, Loss: 30872181607.5549
Epoch 66/200, Loss: 30826690854.3482
Epoch 67/200, Loss: 30770422939.9986
Epoch 68/200, Loss: 30729267469.4737
Epoch 69/200, Loss: 30682508319.3300
Epoch 70/200, Loss: 30643457982.8525
Epoch 71/200, Loss: 30593810134.4537
Epoch 72/200, Loss: 30524204721.2604
Epoch 73/200, Loss: 30509852153.8998
Epoch 74/200, Loss: 30470291754.4939
Epoch 75/200, Loss: 30424528434.0451
Epoch 76/200, Loss: 30387268700.1541
Epoch 77/200, Loss: 30340298372.7232
Epoch 78/200, Loss: 30292553783.1977
Epoch 79/200, Loss: 30266402423.4568
Epoch 80/200, Loss: 30234942103.0829
```

```
Epoch 81/200, Loss: 30196265431.4309
Epoch 82/200, Loss: 30171397634.6651
Epoch 83/200, Loss: 30116929958.8664
Epoch 84/200, Loss: 30085277368.3674
Epoch 85/200, Loss: 30040746622.4453
Epoch 86/200, Loss: 30024261664.6626
Epoch 87/200, Loss: 29985541564.7796
Epoch 88/200, Loss: 29959768930.2246
Epoch 89/200, Loss: 29921983597.5662
Epoch 90/200, Loss: 29892857913.5667
Epoch 91/200, Loss: 29867183607.3532
Epoch 92/200, Loss: 29834513482.5050
Epoch 93/200, Loss: 29781556054.1427
Epoch 94/200, Loss: 29771839093.5616
Epoch 95/200, Loss: 29739059561.2724
Epoch 96/200, Loss: 29707250262.8831
Epoch 97/200, Loss: 29676803227.1695
Epoch 98/200, Loss: 29644603819.1306
Epoch 99/200, Loss: 29626645706.1941
Epoch 100/200, Loss: 29603258124.3484
Epoch 101/200, Loss: 29567050497.6879
Epoch 102/200, Loss: 29539035679.0931
Epoch 103/200, Loss: 29513214848.6663
Epoch 104/200, Loss: 29493903898.7105
Epoch 105/200, Loss: 29454314831.8057
Epoch 106/200, Loss: 29430159726.2473
Epoch 107/200, Loss: 29424373063.9880
Epoch 108/200, Loss: 29381122017.9137
Epoch 109/200, Loss: 29348449087.7557
Epoch 110/200, Loss: 29320619976.0916
Epoch 111/200, Loss: 29303666189.0887
Epoch 112/200, Loss: 29272483036.1985
Epoch 113/200, Loss: 29247377969.4529
Epoch 114/200, Loss: 29223644428.0523
Epoch 115/200, Loss: 29209586519.9195
Epoch 116/200, Loss: 29183348840.2360
Epoch 117/200, Loss: 29139145931.4082
Epoch 118/200, Loss: 29135166545.1382
Epoch 119/200, Loss: 29107102427.3101
Epoch 120/200, Loss: 29086962299.1287
Epoch 121/200, Loss: 29057674916.5862
Epoch 122/200, Loss: 29038945318.2593
Epoch 123/200, Loss: 29011165745.0087
Epoch 124/200, Loss: 28990837969.0642
Epoch 125/200, Loss: 28962861630.9562
Epoch 126/200, Loss: 28946528662.1353
Epoch 127/200, Loss: 28924303730.1562
Epoch 128/200, Loss: 28904584696.3600
Epoch 129/200, Loss: 28868320702.2010
```

```
Epoch 130/200, Loss: 28844010339.7645
Epoch 131/200, Loss: 28840833157.1378
Epoch 132/200, Loss: 28813466992.7348
Epoch 133/200, Loss: 28793033746.4782
Epoch 134/200, Loss: 28765890531.5720
Epoch 135/200, Loss: 28748758923.4452
Epoch 136/200, Loss: 28732438761.2280
Epoch 137/200, Loss: 28702418234.1293
Epoch 138/200, Loss: 28679448885.6282
Epoch 139/200, Loss: 28664186678.1613
Epoch 140/200, Loss: 28644988964.8379
Epoch 141/200, Loss: 28626339905.1475
Epoch 142/200, Loss: 28605650159.2097
Epoch 143/200, Loss: 28571881812.6621
Epoch 144/200, Loss: 28527736665.4593
Epoch 145/200, Loss: 28541871750.6184
Epoch 146/200, Loss: 28520675517.2831
Epoch 147/200, Loss: 28499063331.2389
Epoch 148/200, Loss: 28472708776.9689
Epoch 149/200, Loss: 28450878102.8460
Epoch 150/200, Loss: 28436053741.6699
Epoch 151/200, Loss: 28419608551.7178
Epoch 152/200, Loss: 28407151583.6632
Epoch 153/200, Loss: 28378697318.1631
Epoch 154/200, Loss: 28355934003.7923
Epoch 155/200, Loss: 28349649683.9292
Epoch 156/200, Loss: 28312513265.1049
Epoch 157/200, Loss: 28312868644.1569
Epoch 158/200, Loss: 28278537767.2662
Epoch 159/200, Loss: 28265913147.2546
Epoch 160/200, Loss: 28247283380.5770
Epoch 161/200, Loss: 28220998833.0827
Epoch 162/200, Loss: 28203596721.9415
Epoch 163/200, Loss: 28189397425.7638
Epoch 164/200, Loss: 28170266371.7016
Epoch 165/200, Loss: 28147006799.6872
Epoch 166/200, Loss: 28127371977.1873
Epoch 167/200, Loss: 28117334963.3629
Epoch 168/200, Loss: 28100520284.4798
Epoch 169/200, Loss: 28068883112.3766
Epoch 170/200, Loss: 28058731526.5147
Epoch 171/200, Loss: 28044438479.3171
Epoch 172/200, Loss: 28028001413.3747
Epoch 173/200, Loss: 27997608354.0099
Epoch 174/200, Loss: 27980409823.9593
Epoch 175/200, Loss: 27970330838.1575
Epoch 176/200, Loss: 27952179043.8829
Epoch 177/200, Loss: 27931933804.2633
Epoch 178/200, Loss: 27916620783.2986
```

```
Epoch 179/200, Loss: 27890884740.9009
Epoch 180/200, Loss: 27869207456.8847
Epoch 181/200, Loss: 27865863114.3422
Epoch 182/200, Loss: 27841056736.1962
Epoch 183/200, Loss: 27834448290.1284
Epoch 184/200, Loss: 27804791484.8685
Epoch 185/200, Loss: 27781777504.6552
Epoch 186/200, Loss: 27773487115.6081
Epoch 187/200, Loss: 27750819416.6598
Epoch 188/200, Loss: 27739709176.0935
Epoch 189/200, Loss: 27725987896.9744
Epoch 190/200, Loss: 27713659278.8211
Epoch 191/200, Loss: 27696283876.3123
Epoch 192/200, Loss: 27674252503.9343
Epoch 193/200, Loss: 27663329718.7387
Epoch 194/200, Loss: 27646212756.2401
Epoch 195/200, Loss: 27628351664.5201
Epoch 196/200, Loss: 27619000051.8293
Epoch 197/200, Loss: 27592772053.0619
Epoch 198/200, Loss: 27582322220.1226
Epoch 199/200, Loss: 27563859383.8047
Epoch 200/200, Loss: 27542088791.5345
```



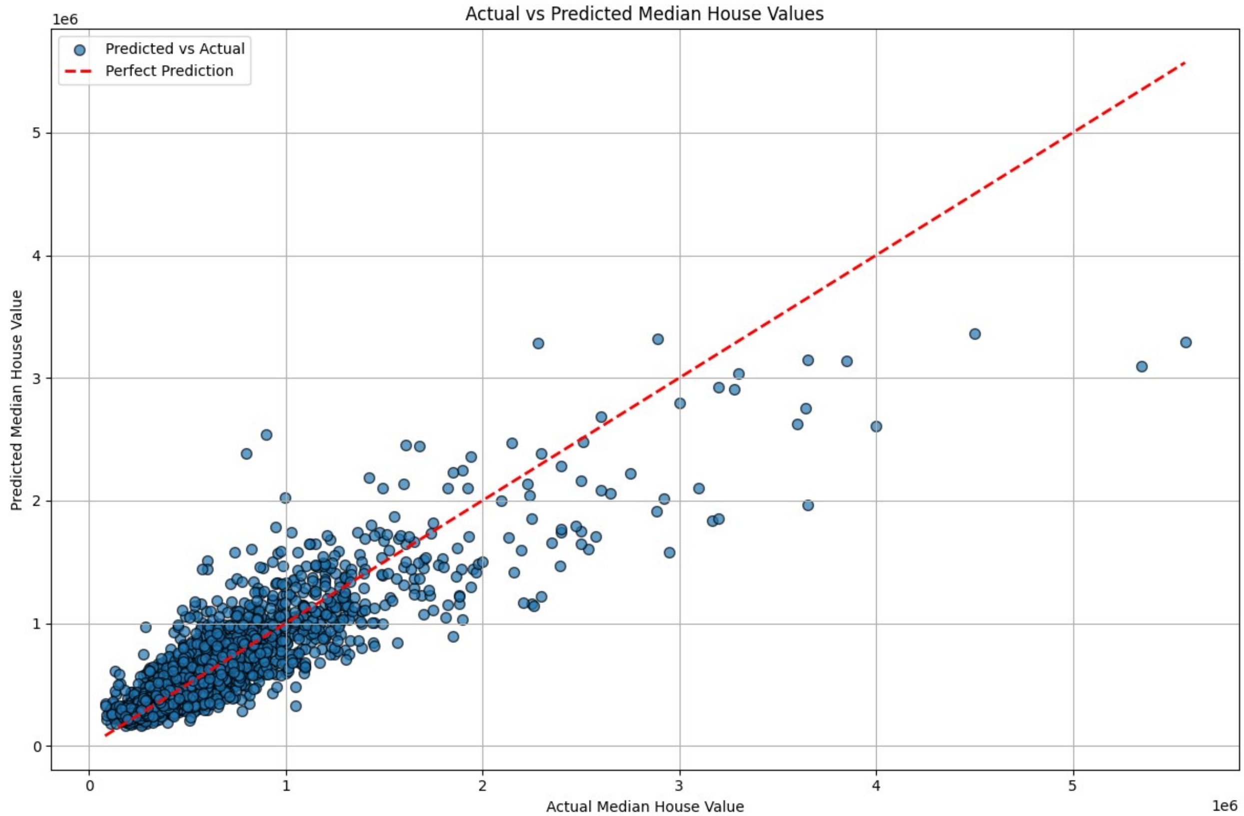
```
test_loss, y_true, y_pred = evaluate_model(model, test_loader,
criterion)
print(f'Test MSE: {test_loss:.4f}')


y_pred = np.vstack(y_pred)
y_test_np = y_test.numpy()


r2 = r2_score(y_test_np, y_pred)
print(f'Test R^2 Score: {r2:.4f}')


plt.figure(figsize=(12, 8))
plt.scatter(y_test_np, y_pred, alpha=0.7, s=50, edgecolors='k',
label='Predicted vs Actual')
plt.xlabel('Actual Median House Value')
plt.ylabel('Predicted Median House Value')
plt.title('Actual vs Predicted Median House Values')
plt.plot([y_test_np.min(), y_test_np.max()], [y_test_np.min(),
y_test_np.max()], 'r--', lw=2, label='Perfect Prediction')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

```
Test MSE: 32483770487.4430
Test R^2 Score: 0.7851
```



Actual vs Predicted Median House Values
1e6
Predicted vs Actual
Perfect Prediction
Predicted Median House Value
0
1
2
3
4
5
Actual Median House Value
1e6