Experiment 3: Keras Regression - Housing Prices Prediction

Problem Statement:

To build an ANN model for regression problem on house predication dataset.

GitHub & Colab Links:

GitHub Link: https://github.com/piyush-gambhir/ncu-lab-manual-and-end-semester-projects/blob/main/NCU-CSL312%20-%20DL%20-%20Lab%20Manual/Experiment%203/Experiment%203.ipynb

Google Colab Link:

Dataset

Dataset Link: https://www.kaggle.com/harlfoxem/housesalesprediction

Dataset Description

This dataset contains house sale prices for King County, which includes Seattle. It includes homes sold between May 2014 and May 2015.

Feature Columns

- id - Unique ID for each home sold
- date - Date of the home sale
- price - Price of each home sold
- bedrooms - Number of bedrooms
- bathrooms - Number of bathrooms, where .5 accounts for a room with a toilet but no shower
- sqft_living - Square footage of the apartments interior living space
- sqft_lot - Square footage of the land space
- floors - Number of floors
- waterfront - A dummy variable for whether the apartment was overlooking the waterfront or not
- view - An index from 0 to 4 of how good the view of the property was
- condition - An index from 1 to 5 on the condition of the apartment,
- grade - An index from 1 to 13, where 1-3 falls short of building construction and design, 7 has an average level of construction and design, and 11-13 have a high quality level of construction and design.
- sqft_above - The square footage of the interior housing space that is above ground level
- sqft_basement - The square footage of the interior housing space that is below ground level
- yr_built - The year the house was initially built
- yr_renovated - The year of the house's last renovation
- zipcode - What zipcode area the house is in
- lat - Lattitude
- long - Longitude
- sqft_living15 - The square footage of interior housing living space for the nearest 15 neighbors
- sqft_lot15 - The square footage of the land lots of the nearest 15 neighbors

Code

In [ ]:
```
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
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_squared_error, mean_absolute_error, explained_variance_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.optimizers import Adam
```

In [ ]:
```
# Load the Dataset
df = pd.read_csv('kc_house_data.csv')
```

In [ ]:
```
# Preliminary Data Exploration
print(df.isnull().sum())  # Check for null values
print(df.describe().transpose())  # Summary statistics
```

```
id               0
date             0
price            0
bedrooms         0
bathrooms        0
sqft_living      0
sqft_lot         0
floors           0
waterfront       0
view             0
condition        0
grade            0
sqft_above       0
sqft_basement    0
yr_built         0
yr_renovated     0
zipcode          0
lat              0
long             0
sqft_living15    0
sqft_lot15       0
dtype: int64
                 count          mean           std           min  \
id             21597.0  4.580474e+09  2.876736e+09  1.000102e+06
price          21597.0  5.402966e+05  3.673681e+05  7.800000e+04
bedrooms       21597.0  3.373200e+00  9.262989e-01  1.000000e+00
bathrooms      21597.0  2.115826e+00  7.689843e-01  5.000000e-01
sqft_living    21597.0  2.080322e+03  9.181061e+02  3.700000e+02
sqft_lot       21597.0  1.509941e+04  4.141264e+04  5.200000e+02
floors         21597.0  1.494096e+00  5.396828e-01  1.000000e+00
waterfront     21597.0  7.547345e-03  8.654900e-02  0.000000e+00
view           21597.0  2.342918e-01  7.663898e-01  0.000000e+00
condition      21597.0  3.409825e+00  6.505456e-01  1.000000e+00
grade          21597.0  7.657915e+00  1.173200e+00  3.000000e+00
sqft_above     21597.0  1.788597e+03  8.277598e+02  3.700000e+02
sqft_basement  21597.0  2.917250e+02  4.426678e+02  0.000000e+00
yr_built       21597.0  1.971000e+03  2.937523e+01  1.900000e+03
yr_renovated   21597.0  8.446479e+01  4.018214e+02  0.000000e+00
zipcode        21597.0  9.807795e+04  5.351307e+01  9.800100e+04
lat            21597.0  4.756009e+01  1.385518e-01  4.715590e+01
long           21597.0 -1.222140e+02  1.407235e-01 -1.225190e+02
sqft_living15  21597.0  1.986620e+03  6.852305e+02  3.990000e+02
sqft_lot15     21597.0  1.275828e+04  2.727444e+04  6.510000e+02

                        25%           50%           75%           max
id             2.123049e+09  3.904930e+09  7.308900e+09  9.900000e+09
price          3.220000e+05  4.500000e+05  6.450000e+05  7.700000e+06
bedrooms       3.000000e+00  3.000000e+00  4.000000e+00  3.300000e+01
bathrooms      1.750000e+00  2.250000e+00  2.500000e+00  8.000000e+00
sqft_living    1.430000e+03  1.910000e+03  2.550000e+03  1.354000e+04
sqft_lot       5.040000e+03  7.618000e+03  1.068500e+04  1.651359e+06
floors         1.000000e+00  1.500000e+00  2.000000e+00  3.500000e+00
waterfront     0.000000e+00  0.000000e+00  0.000000e+00  1.000000e+00
view           0.000000e+00  0.000000e+00  0.000000e+00  4.000000e+00
condition      3.000000e+00  3.000000e+00  4.000000e+00  5.000000e+00
grade          7.000000e+00  7.000000e+00  8.000000e+00  1.300000e+01
sqft_above     1.190000e+03  1.560000e+03  2.210000e+03  9.410000e+03
sqft_basement  0.000000e+00  0.000000e+00  5.600000e+02  4.820000e+03
yr_built       1.951000e+03  1.975000e+03  1.997000e+03  2.015000e+03
yr_renovated   0.000000e+00  0.000000e+00  0.000000e+00  2.015000e+03
zipcode        9.803300e+04  9.806500e+04  9.811800e+04  9.819900e+04
lat            4.747110e+01  4.757180e+01  4.767800e+01  4.777760e+01
long          -1.223280e+02 -1.222310e+02 -1.221250e+02 -1.213150e+02
sqft_living15  1.490000e+03  1.840000e+03  2.360000e+03  6.210000e+03
sqft_lot15     5.100000e+03  7.620000e+03  1.008300e+04  8.712000e+05
```

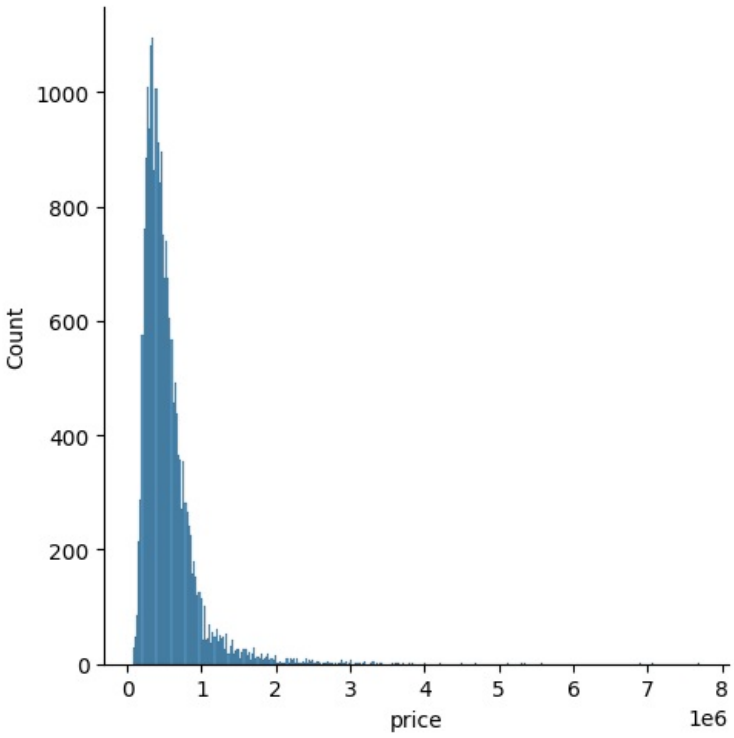
In [ ]:
```
# Visualizing the Data
print("Visualizing the Data")

plt.figure(figsize=(12, 8))
sns.displot(df['price'])  # Distribution of house prices

plt.figure(figsize=(12, 8))
sns.countplot(df['bedrooms'])  # Count of bedrooms

plt.figure(figsize=(12, 8))
sns.scatterplot(x='price', y='sqft_living', data=df)  # Price vs. living area

plt.figure(figsize=(12, 8))
```

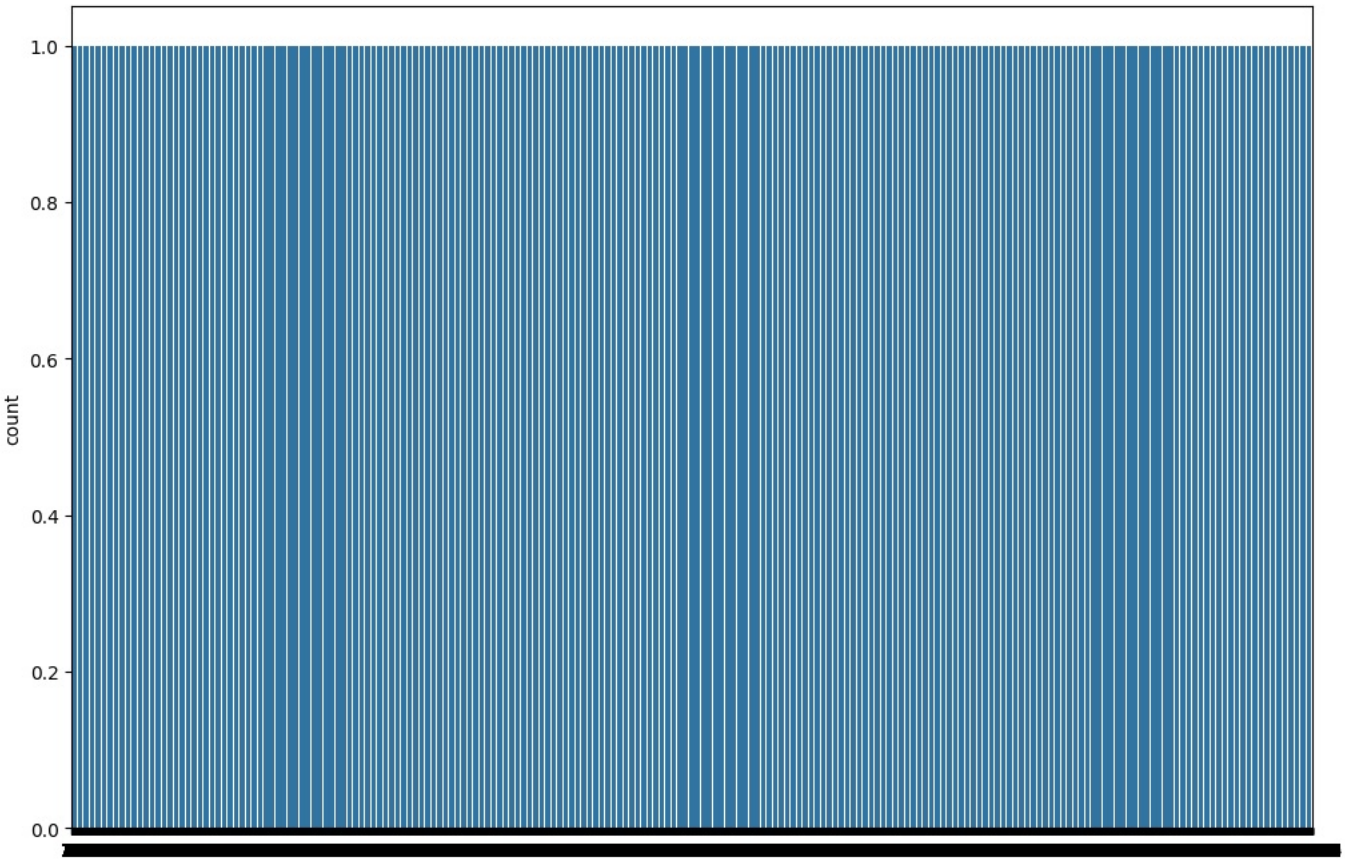
```
sns.boxplot(x='bedrooms', y='price', data=df)  # Price by bedroom count
```

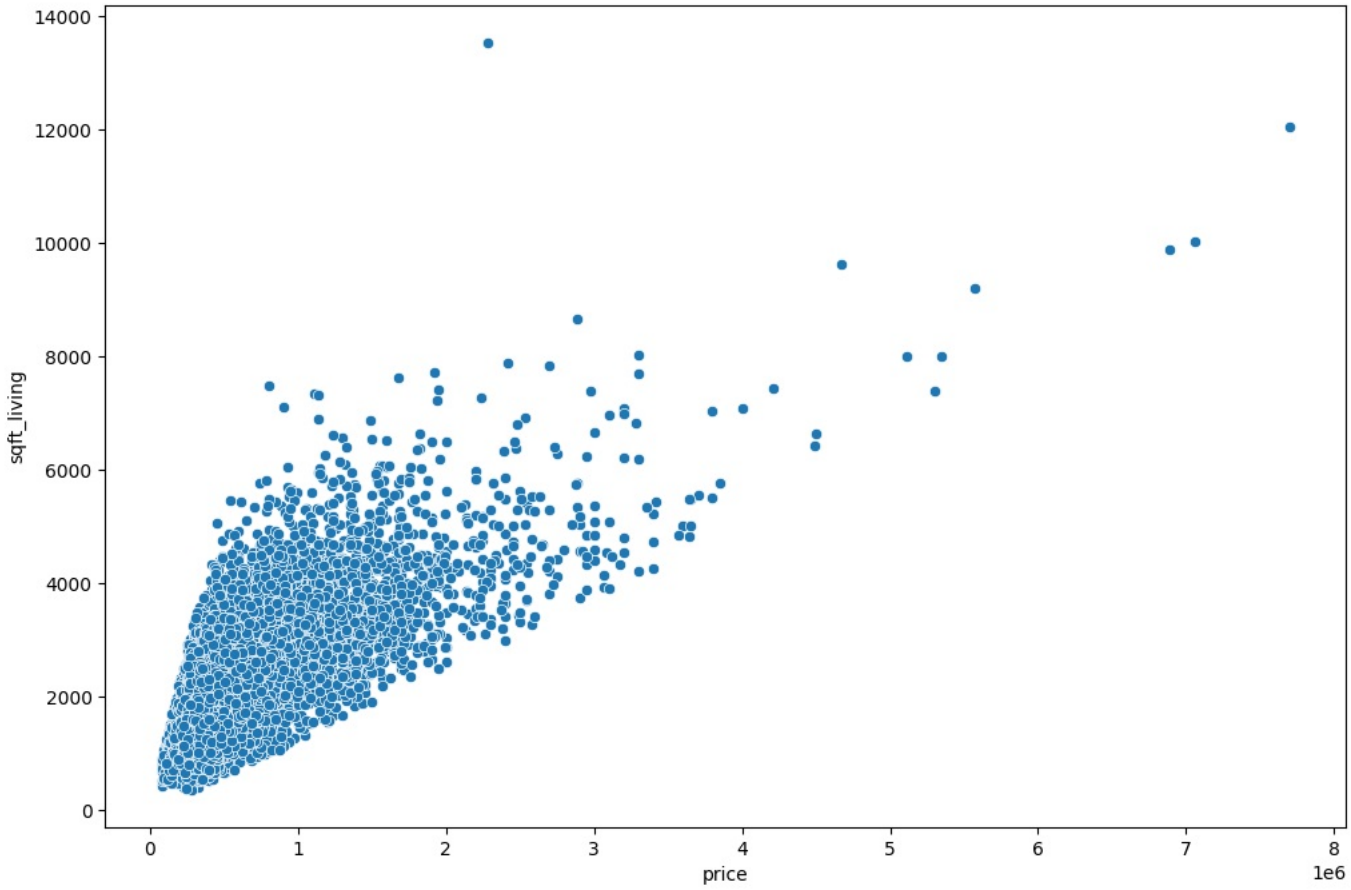
Visualizing the Data

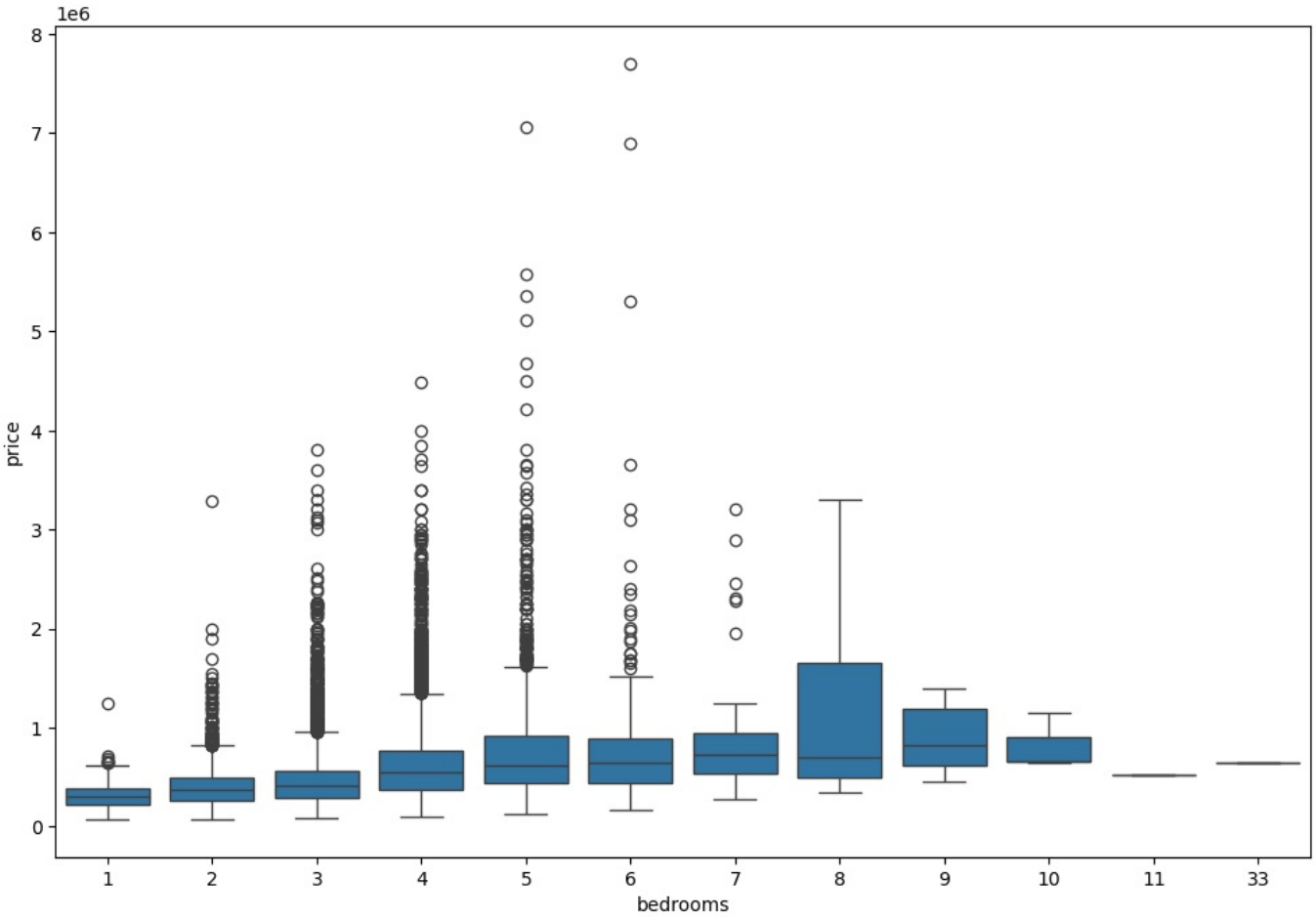
Out[ ]: <Axes: xlabel='bedrooms', ylabel='price'>

<Figure size 1200x800 with 0 Axes>






14000
12000
10000
8000
6000
4000
2000
0
sqft_living
0
1
2
3
4
5
6
7
8
price
1e6



In [ ]:
```python
# Dropping unnecessary features
df.drop(['id', 'date', 'zipcode'], axis=1, inplace=True)

# Feature Engineering from 'yr_renovated' and 'sqft_basement'
df['yr_renovated'] = df['yr_renovated'].apply(lambda x: 1 if x > 0 else 0)
df['has_basement'] = df['sqft_basement'].apply(lambda x: 1 if x > 0 else 0)
df.drop('sqft_basement', axis=1, inplace=True)

# Scaling and Train Test Split
X = df.drop('price', axis=1).values
y = df['price'].values
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.3, random_state=101)

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

In [ ]:
```
# Creating the Neural Network Model
model = Sequential([
    Dense(19, activation='relu'),
    Dense(19, activation='relu'),
    Dense(19, activation='relu'),
    Dense(19, activation='relu'),
    Dense(1)
])

model.compile(optimizer=Adam(), loss='mse')

# Training the Model
model.fit(x=X_train, y=y_train, validation_data=(
    X_test, y_test), batch_size=128, epochs=400, verbose=0)
```

Out[ ]:
```
<keras.src.callbacks.history.History at 0x19e069d79d0>
```

In [ ]:
```
# Evaluating Model Performance
losses = pd.DataFrame(model.history.history)
losses.plot()

predictions = model.predict(X_test)
print(f"MAE: {mean_absolute_error(y_test, predictions)}")
print(f"RMSE: {np.sqrt(mean_squared_error(y_test, predictions))}")
print(
    f"Explained Variance Score: {explained_variance_score(y_test, predictions)}")

# Plotting predictions vs actual prices
plt.scatter(y_test, predictions)
plt.plot(y_test, y_test, 'r')
```

```
203/203 ━━━━━━━━━━━━━━━━━━━━ 0s 935us/step
MAE: 100499.83408323688
RMSE: 164049.0437463553
Explained Variance Score: 0.7973987069435204
```

Out[ ]:
```
[<matplotlib.lines.Line2D at 0x19e1164d050>]
```