Artificial Intelligence Midterm Project

In this project, you will build a regression model and a classification model from scratch. Please follow the instructions closely, and only use Python's Numpy, Pandas, and matplotlib library to complete this project. Using functions from `sklearn` is not allowed.

Part I dues on Monday, March 22nd at 11:59 PM. **Part II** dues on Monday, April 12th at 11:59 PM.

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

Part I: A Regression Model

In this part, please build a multilinear regression model that extracts the relationship between housing prices and other relevant variables. The training data is shown in the table below:

In [2]:
```
data1 = pd.DataFrame({
    "YearBuilt": [1974, 1996, 1968, 1962, 1960],
    "YearSold": [2015, 2017, 2020, 2010, 2016],
    "Bedrooms": [3, 10, 4, 5, 6],
    "TotalArea": [1500, 4000, 1700, 2500, 2000],
    "Quality": [7.5, 6, 4, 5.5, 5],
    "Price": [358500, 452600, 352100, 341300, 342200]
})

data1
```

Out[2]:

	YearBuilt	YearSold	Bedrooms	TotalArea	Quality	Price
0	1974	2015	3	1500	7.5	358500
1	1996	2017	10	4000	6.0	452600
2	1968	2020	4	1700	4.0	352100
3	1962	2010	5	2500	5.5	341300
4	1960	2016	6	2000	5.0	342200

Task 1: Data Transformation (10 pts)

Create a new column named "Age" that represents the age of each house when it was sold.

In [3]:
```
# Your Code Here
def ageOfHome(YearBuilt, YearSold):
    return YearSold - YearBuilt
```

In [6]:
```
data1["Age"] = ageOfHome(data1["YearBuilt"], data1["YearSold"])
data1["Age"]
```

Out[6]:
```
0    41
1    21
2    52
3    48
4    56
Name: Age, dtype: int64
```

In [ ]:
```

```

Task 2: Train a Multilinear Model (20 pts)

Assume that the price can be expressed as a linear combination of age, bedrooms, total area, and quality:

$$Price = \theta_0 + \theta_1 \cdot Age + \theta_2 \cdot Bedrooms + \theta_3 \cdot TotalArea + \theta_4 \cdot Quality.$$

Apply the normal equation to find the best values for the parameters:

1. Construct matrix $\mathbf{X}$ and $\mathbf{y}$ (the matrices are defined in Week 6 notebook and Chapter 4 of the textbook).
2. Calculate the parameter vector using the normal equation $\theta = \left(\mathbf{X}^T \cdot \mathbf{X}\right)^{-1} \cdot \mathbf{X}^T \cdot \mathbf{y}$

In [15]:
```
# Construct X and y as numpy arrays
X = np.hstack([np.ones([len(data1), 1]), data1[['Age', 'Bedrooms', 'TotalArea', 'Quality' ]].values])
print(X)
print("_____________________________________________________________")
```

```
[[1.0e+00 4.1e+01 3.0e+00 1.5e+03 7.5e+00]
 [1.0e+00 2.1e+01 1.0e+01 4.0e+03 6.0e+00]
 [1.0e+00 5.2e+01 4.0e+00 1.7e+03 4.0e+00]
 [1.0e+00 4.8e+01 5.0e+00 2.5e+03 5.5e+00]
 [1.0e+00 5.6e+01 6.0e+00 2.0e+03 5.0e+00]]
_____________________________________________________________
```

In [16]:
```
y = data1[["Price"]].values
print(y)
```

```
[[358500]
 [452600]
 [352100]
 [341300]
 [342200]]
```

In [17]:
```
theta = np.linalg.inv(X.T.dot(X)).dot(X.T).dot(y)
print(theta)
```

```
[[ 5.92376387e+05]
 [-3.83925328e+03]
 [ 1.17271948e+04]
 [-3.11089808e+01]
 [-8.66468214e+03]]
```

Task 3: Make A Prediction (10 pts)

Suppose that there is another house with the following attribute:

- YearBuilt: 1985
- YearSold: 2021
- Bedrooms: 6
- Total Area: 2500
- Quality: 5.5

Use the parameter values that you have calculated to make a prediction on its sale price.

In [19]:
```
# Your Code Here
theta0 =  5.92376387
theta1 = -3.83925328
theta2 = 1.17271948
theta3 = -3.11089808
theta4 = -8.66468214

data1["Price"] = theta0 + theta1 * data1["Age"] + theta2 * data1["Bedrooms"] + theta3 * data1["TotalArea"] + the

print(data1["Price"])
```

```
0    -4879.299698
1   -12558.553773
2    -5512.211993
3    -7997.397748
4    -6467.157674
Name: Price, dtype: float64
```

In [18]:
```
theta0 =  5.92376387
theta1 = -3.83925328
theta2 = 1.17271948
theta3 = -3.11089808
theta4 = -8.66468214

prediction = theta0 + theta1 * 36 + theta2 * 6 + theta3 * 2500 + theta4 * 5.5
print(prediction)
```

```
-7950.1539891
```

Part II: A Classification Model

In this part, we will build a logistic regression model and evaluate its performance on the classifying the data. The dataset is as follows:

In [ ]:
```
data2 = pd.DataFrame([[5.0, 2.0, 1],
                      [6.2, 3.4, 1],
                      [4.9, 3.6, 0],
                      [6.2, 2.2, 1],
                      [5.7, 3.0, 1],
                      [4.8, 3.4, 0],
                      [5.0, 3.4, 0]],
                     columns=["x1", "x2", "class"])
data2
```

Out[16]:

	x	y	class
0	5.0	2.0	1
1	6.2	3.4	1
2	4.9	3.6	0
3	6.2	2.2	1
4	5.7	3.0	1
5	4.8	3.4	0
6	5.0	3.4	0

Task 1: Data Visualization (10 pts)

Visualize the data as a scatter plot. Show class 0 records as green dots and class 1 records as blue dots. Display the following items:

- Title of the plot: Distribution of the training data
- Label for x axis: x1
- Label for y axis: x2
- Legend

In [ ]:
```
# Your Code Here
```

Task 2: Apply A Logistic Regression Model (10 pts)

Suppose that you are given a logistic regression model with explicity paramter values:

$p = \sigma(\mathbf{x} \cdot \theta^T)$.

where

- p: the probability that the point belongs to class 1.
- $\mathbf{x} = (1, x_1, x_2)$.
- $\theta = (\theta_0, \theta_1, \theta_2) = (-2.15, 0.92, -0.82)$.
- $\sigma(t) = \frac{1}{1+e^{-t}}$

Find the model's prediction on the following test set:

In [ ]:
```
data3 = pd.DataFrame([[5.1, 3.4, 0],
                      [6.5, 2.8, 1],
                      [5.8, 2.7, 1],
                      [4.6, 3.1, 0]],
                     columns=["x1", "x2", "class"])
data3
```

Out[22]:

	x1	x2	class
0	5.1	3.4	0
1	6.5	2.8	1
2	5.8	2.7	1
3	4.6	3.1	0

Task 3: Model Evaluation (40 pts)

Calculate the following model metrics regarding the performance on the test set:

- classification accuracy
- precision score
- recall score
- F-1 score

In [ ]:
```
# Classification Accuracy
```

In [ ]:
```
# Precision Score
```

In [ ]:
```
# Recall Score
```

In [ ]:
```
# F-1 Score
```

In [ ]:
```

```