# Assignment

## Homework 01

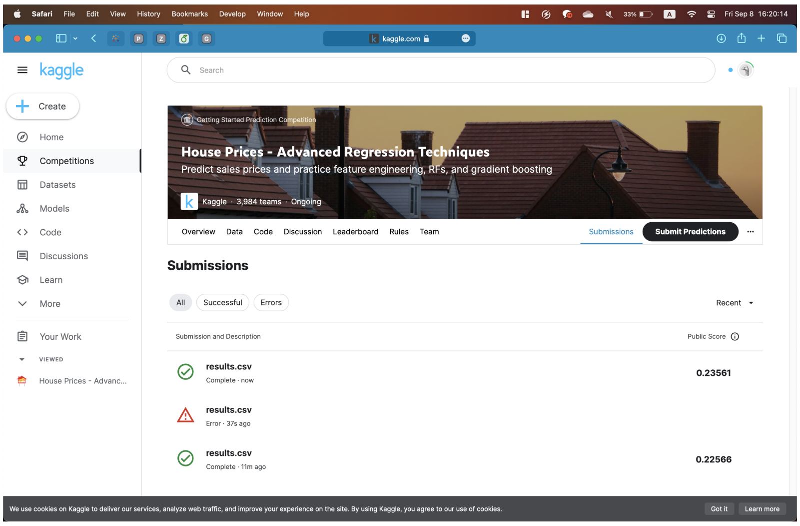
**A**ndrew Park

Applied Machine Learning Homework Assignment

CORNELL TECH | HOME OF THE JACOBS TECHNION-CORNELL INSTITUTE

September 9, 2023

## REPORT for Part II. The Housing Prices




1. Initially, the predicted values came out too low, so I made some code modifications and uploaded it again. However, it seems that the data I chose didn't have a significant impact on housing prices, as the predictions continue to come out the same.

2. I selected the following features for my house price prediction model to capture both the structural aspects of a property (age and size) and the local zoning regulations that influence property values:

   (a) YearBuilt: The age of a building can significantly impact its price. Newer buildings often have higher prices due to modern amenities and construction standards.

   (b) 1stFlrSF and 2ndFlrSF: The total square footage of the first and second floors are crucial indicators of a home's size and living space. Larger living spaces typically command higher prices.

   (c) GrLivArea: The total above-ground living area is a critical factor in determining a home's price. Larger living areas are generally associated with higher prices.

   (d) MSZoning: The zoning classification of a property can significantly impact its value. Different zones have specific regulations and restrictions that can affect property values. Understanding the zoning classification can provide valuable insights into the price of a property.

3. This was based on my assumption that zoning data would be important due to my past experience as an architect. However, in the given dataset, it appears that zoning data is not strongly correlated with housing prices, which resulted in poor predictions.

```python
import pandas as pd
from sklearn.preprocessing import StandardScaler

# Read the CSV file into a DataFrame
test_df = pd.read_csv("test.csv")

# Select specific columns
columns_to_select = ['YearBuilt', '1stFlrSF', '2ndFlrSF', 'GrLivArea', 'MSZoning']
test_df_set = test_df[columns_to_select].copy()  # Create a copy of the DataFrame

# Select numeric columns for scaling
numeric_columns = test_df_set.select_dtypes(include=['int64', 'float64']).columns

# Initialize the StandardScaler
scaler = StandardScaler()

# Fit and transform the selected numeric columns
test_df_set[numeric_columns] = scaler.fit_transform(test_df_set[numeric_columns])

# Check if the 'SaleType' column is of object dtype (categorical)
if test_df_set['MSZoning'].dtype == 'object':
    test_dummies = pd.get_dummies(test_df_set['MSZoning'], prefix='MSZoning', dtype=int)
    test_df_set.drop(columns=['MSZoning'], inplace=True)  # Remove the original column
    test_df_set = pd.concat([test_df_set, test_dummies], axis=1)  # Concatenate the dummy variables

test_df_set
```

✓ 0.0s

| | YearBuilt | 1stFlrSF | 2ndFlrSF | GrLivArea | MSZoning_C (all) | MSZoning_FV | MSZoning_RH | MSZoning_RL | MSZoning_RM |
|---|---|---|---|---|---|---|---|---|---|
| 0 | -0.340945 | -0.654561 | -0.775254 | -1.215588 | 0 | 0 | 1 | 0 | 0 |
| 1 | -0.439695 | 0.433298 | -0.775254 | -0.323539 | 0 | 0 | 0 | 1 | 0 |
| 2 | 0.844059 | -0.574165 | 0.891944 | 0.294508 | 0 | 0 | 0 | 1 | 0 |
| 3 | 0.876976 | -0.579190 | 0.837243 | 0.243004 | 0 | 0 | 0 | 1 | 0 |
| 4 | 0.679475 | 0.310192 | -0.775254 | -0.424487 | 0 | 0 | 0 | 1 | 0 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1454 | -0.044694 | -1.533893 | 0.523306 | -0.811797 | 0 | 0 | 0 | 0 | 1 |
| 1455 | -0.044694 | -1.533893 | 0.523306 | -0.811797 | 0 | 0 | 0 | 0 | 1 |
| 1456 | -0.373861 | 0.169499 | -0.775254 | -0.539856 | 0 | 0 | 0 | 1 | 0 |
| 1457 | 0.679475 | -0.468645 | -0.775254 | -1.063136 | 0 | 0 | 0 | 1 | 0 |
| 1458 | 0.712392 | -0.403324 | 1.612573 | 1.058827 | 0 | 0 | 0 | 1 | 0 |

1459 rows × 9 columns

```python
test_df_set = test_df_set[X.columns]

reg.fit(X, y)
pred = reg.predict(test_df_set)

# Create a DataFrame for the predictions
pred_df = pd.DataFrame(pred, columns=['SalePrice'])

# Combine 'Id' with predicted values
result = pd.concat([test_df['Id'], pred_df], axis=1)
result['Id'] = result['Id'].astype(int)

result

result.to_csv('results.csv', index=False)
```

✓ 0.0s

4. The low prediction values were not solely due to the creation of dummy variables for ’MSZoning,’ which resulted in over half of the training data (5 columns) being transformed into random values. Additionally, the limited presence of only 4 columns with normalized numeric values could have contributed to this issue. In conclusion, for future prediction tasks, it appears advantageous to work with datasets consisting entirely of numeric values or to convert non-numeric data into a numerical format based on their category (e.g., 0, 1, 2, 3, etc.) rather than relying on random values. By adopting this approach, we can maintain the inherent characteristics of the data columns, ultimately leading to more accurate predictions.

## WRITTEN EXERCISES

### Problem 1

For each of the problem, identify whether it's more naturally characterized as a binary classification, multiclass classification, multilabel classification, regression, clustering, density modeling, or RL problem.

**Solution.**

a. Given a stream of customers each characterized by some attributes, learn which ads to show them given that you can only show each customer one ad.
**Multiclass Classification**: Each ads can be different classes, and one class (ad) can be shown to one customer.

b. Classify emails as spam or not spam.
**Binary Classification**: The mail can be either classified 1(spam) or 0(not spam).

c. Given a news article, predict which topics it covers.
**Multilabel Classification**: News topics can have multiple overlapping topics.

d. Given a pair of images of faces, identify whether they depict the same person.
**Binary Classification**: The image can be either classified 1(same person) or 0(different person).

e. Learn to play a board game against randomly matched opponents on the internet.
**RL**: Learning the optimal strategy to win the game.

f. Identify whether a new data point is expected given those you have seen before or extremely unlikely.
**Density Modeling**: Involves estimation of probability distribution based on the data points you have seen before

g. Figure out whether a group of patients happens to naturally break down into some number of subgroups.
**Clustering**: Grouping patients based on their characteristics, which identifies natural subgroups.

h. Recognize celebrities based on photographs scraped from Twitter.
**Multiclass Classification**: Classifying images into celebrities(classes)

i. Predict the starting salaries of new graduates based on their academic record.
**Regression**: Predicting the salary(continuous variable) based on the academic record

j. Identify the best (personalized) treatment among a set of drugs for a given chronic condition.
**Multiclass Classification**: Each drugs(treatment) can be different classes, and each patient can be classified to a specific treatment.

**Problem 2**

Based on the materials covered so far, in supervised machine learning, why must we make assumptions? Why can't we just learn from data alone?

**Solution.**

- Without strong assumptions, the existing data is not informative of unseen future data. It is important in machine learning because assumptions help setting solution space and boundaries.

**Problem 3**

Analytical solution of the Ordinary Least Squares Estimation.

**Solution.**

a. $\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$

$= \frac{\partial}{\partial \theta_0} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2$

$= \frac{\partial}{\partial \theta_0} \sum_{i=1}^{n} (y^{(i)} - \theta_0 - \theta_1 x^{(i)})^2$

$= -2 \sum_{i=1}^{n} (y^{(i)} - \theta_0 - \theta_1 x^{(i)})$

$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$

$= \frac{\partial}{\partial \theta_1} \sum_{i=1}^{n} (y^{(i)} - \hat{y}^{(i)})^2$

$= \frac{\partial}{\partial \theta_1} \sum_{i=1}^{n} (y^{(i)} - \theta_0 - \theta_1 x^{(i)})^2$

$= -2 \sum_{i=1}^{n} x^{(i)} (y^{(i)} - \theta_0 - \theta_1 x^{(i)})$

b. By using the information from the previous question, we can derive the two proprieties.

$J(\theta_0, \theta_1)$ has an unique optimum of ${\theta_0}^*, {\theta_1}^*$

$\frac{\partial}{\partial \theta} J({\theta_0}^*, \theta_1) = 0$

$\Rightarrow -2 \sum_{i=1}^{n} (y^{(i)} - \theta_0 - \theta_1 x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} (y^{(i)} - \theta_0 - \theta_1 x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} y^{(i)} - \sum_{i=1}^{n} \theta_0 - \sum_{i=1}^{n} \theta_1 x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} y^{(i)} - n\theta_0 - \theta_1 \sum_{i=1}^{n} x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} y^{(i)} - \theta_1 \sum_{i=1}^{n} x^{(i)}) = n\theta_0$

$\Rightarrow \sum_{i=1}^{n} \frac{y^{(i)}}{n} - \theta_1 \sum_{i=1}^{n} \frac{x^{(i)}}{n} = \theta_0$

$\Rightarrow \therefore \theta_0{}^* = \overline{y} - \theta_1\overline{x}$

$\frac{\partial}{\partial \theta_0} J(\theta_0{}^*, \theta_1) = 0$

$\Rightarrow -2 \sum_{i=1}^{n} x^{(i)}(y^{(i)} - \theta_0 - \theta_1 x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} (x^{(i)}y^{(i)} - x^{(i)}\theta_0 - \theta_1 x^{(i)^2}) = 0$

$\Rightarrow \sum_{i=1}^{n} (x^{(i)}y^{(i)} - x^{(i)}(\overline{y} - \theta_1\overline{x}) - \theta_1 x^{(i)^2}) = 0$

$\Rightarrow \sum_{i=1}^{n} (x^{(i)}y^{(i)} - x^{(i)}\overline{y} + x^{(i)}\theta_1\overline{x} - \theta_1 x^{(i)^2}) = 0$

$\Rightarrow \sum_{i=1}^{n} (x^{(i)}y^{(i)} - x^{(i)}\overline{y}) - \sum_{i=1}^{n} \theta_1 x^{(i)}(-\overline{x} + x^{(i)}) = 0$

$\Rightarrow \sum_{i=1}^{n} x^{(i)}(y^{(i)} - \overline{y}) = \theta_1 \sum_{i=1}^{n} x^{(i)}(x^{(i)} - \overline{x})$

$\therefore \theta_1{}^* = \frac{\sum_{i=1}^{n} x^{(i)}(y^{(i)} - \overline{y})}{\sum_{i=1}^{n} x^{(i)}(x^{(i)} - \overline{x})}$

c. $\sum_{i=1}^{n} e^{(i)} = \sum_{i=1}^{n} (y^{(i)} - (\theta_0{}^* + \theta_1{}^* x^{(i)}))$

$= \sum_{i=1}^{n} (y^{(i)} - (\overline{y} - \theta_1{}^*\overline{x}) - \theta_1{}^* x^{(i)})$

$= \sum_{i=1}^{n} y^{(i)} - \sum_{i=1}^{n} \overline{y} - \sum_{i=1}^{n} \theta_1{}^*\overline{x} - \sum_{i=1}^{n} \theta_1{}^* x^{(i)}$

$= \sum_{i=1}^{n} y^{(i)} - n\overline{y} - n\theta_1{}^*\overline{x} - \theta_1{}^* \sum_{i=1}^{n} x^{(i)}$

$= \sum_{i=1}^{n} y^{(i)} - n\frac{\sum_{i=1}^{n} y^{(i)}}{n} - n\theta_1{}^*\frac{\sum_{i=1}^{n} x^{(i)}}{n} - \theta_1{}^* \sum_{i=1}^{n} x^{(i)}$

$= \sum_{i=1}^{n} y^{(i)} - \sum_{i=1}^{n} y^{(i)} - \theta_1{}^* \sum_{i=1}^{n} x^{(i)} - \theta_1{}^* \sum_{i=1}^{n} x^{(i)}$

$= 0$

- What can be learned from the value of $\sum_{i=1}^{n} e^{(i)}$ ?

  We can learn that with the unique optimum of $\theta_0{}^*$ and $\theta_1{}^*$, the sum of residuals $\sum_{i=1}^{n} e^{(i)}$ equals 0. This means that half of the positive residuals will be exactly balanced by half of the negative residuals, indicating that there is no over- or under-prediction in the model.