Session 3
Homework 2 Introduction

Fall 2025 ECE 157/272A
Seoyeon Kim, Heekyung Lee

Overview

- Familiarize with the process of building machine learning models using Prompt Engineering

- Two datasets, each designated for a specific task
  - California Housing Prices for Regression
  - Dry Bean Classification for Multiclass classification

- The process is divided into three parts:
  - Data Exploration
  - Data Preprocessing
  - Model Building and Evaluation

Data Exploration

Data Structure and Statistic

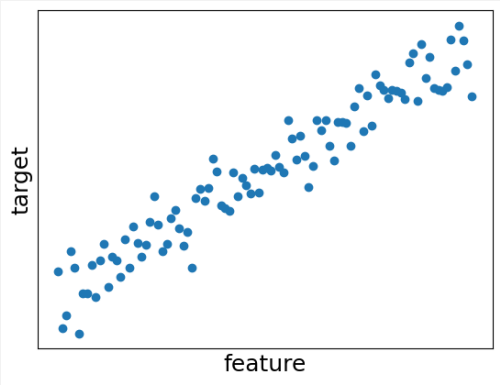
- **Examine the structure of the dataset**
  - Number of rows: number of samples in the dataset
  - Column names and column data types: features that are available and the data types of the features
- **Examine the statistic of the data**
  - Distribution of the data
    - Continuous: value range, density peak, quantiles, mean, standard deviation
    - Categorical: unique values, percentage of each value
  - Identify features that might aid class separation or enhance the prediction of target values
  - Identify any skewness, outlier, or missing values

Data Correlation

➢ Pearson Correlation

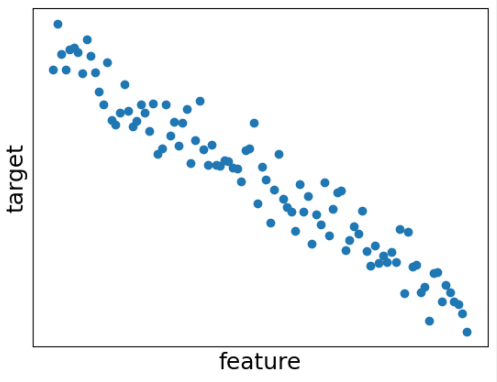
- Correlation coefficient [-1, 1]

- Highly Correlated (Positive Correlation):
  - Correlation coefficient close to 1
  - As one variable increases, the other tends to increase



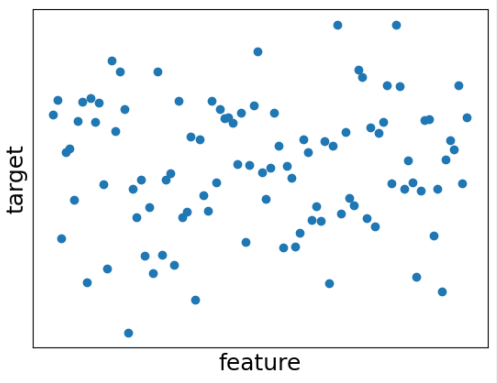
High Correlation

- Highly Negatively Correlated (Negative Correlation):
  - Correlation coefficient close to -1
  - As one variable increases, the other tends to decrease



High Negative Correlation

- No/Low Correlation (Correlation Close to 0):
  - Correlation coefficient close to 0
  - Suggests little to no linear relationship
  - Changes in one variable are not predictably associated with changes in the other



Low/No Correlation

Data Correlation

- **Correlations**
  - **Examine feature-to-target-variable correlation to keep highly correlated features and remove low correlation features**
  - **Examine feature-to-feature correlation to identify highly correlated features pairs that potentially conveys similar information**
    - **Consider dropping one of the highly correlated features (curse of dimensionality)**

Data Preprocessing

Data Preprocessing

- **Prepare the dataset for training and evaluating models**
  - **Splitting the dataset**
  - **Building preprocessing pipeline**

Data Splitting

- **Model training and evaluation consist of three dataset:**
  - **Training, Validation, and Test**

- **The three dataset should have the same data distribution**
  - **Ensure the model's performance on the validation and test set is a meaningful indicator of its ability to handle diverse, unseen data**

Training set

- Train machine learning models
- Model learns from the dataset, learning the pattern and relationship within the data
- Model's internal parameters are adjusted to minimize the difference between the prediction and the ground truth values in the training set

Training set

- **Is model performing well on the training dataset sufficient?**
- **NO! The model could be overfitting to the training dataset**
  - **Learning noisy data or irrelevant features in the training dataset that does not represent the true behavior of the data**
  - **Or simply "memorizing" every training samples**
- **Therefore, we need a separate and unseen dataset to test the trained model**

Validation Set

- **Serves an unseen dataset to test whether the model has learned the true behavior of the data**

- **Under-fitting**
  - **Model performs poorly on training set and validation set**
  - **Model did not learn the behavior**
  - **Could be due to low model complexity or insufficient training iteration**

- **Over-fitting**
  - **Model performs well on training set but poorly on validation set**
  - **Model did not learn the true behavior**
  - **The model has enough complexity to learn the behavior of the training data and even the noise within it**

- **Well-Train and Generalizable**
  - **Model performs well on training set and validation set**
  - **The behavior learned from the training set is applicable to unseen data in the validation set**

Validation Set and Hyperparameters

- **How to fix underfitting and overfitting?**
  - Tune model hyperparameters to adjust the model complexity and retrain
  - Hyperparameters are training and model configurations that can be adjusted by the programmer
    - For example, the decision tree's depth, the number of neighbors in nearest neighbors, the number of training iterations, …

- **After a series of hyperparameter tuning, we have obtained a model that performs well on the training and validation set. Does that mean this model is able to generalize well?**
  - Not quite! Validation set provides feedbacks to perform the hyperparameter tuning which influence the training of the model and the adjustment of the internal parameters
  - We need the Test set to provide another level of generalization testing

Test Set

- Kept separate from both the training and validation sets
- Provides an unbiased evaluation to assess the final performance of the trained model, especially model's generalization to new, previously unseen samples

Data Preprocessing Recap

- **Prepare the dataset for training and evaluating models**
  - **Splitting the dataset**
  - **Building preprocessing pipeline**

Preprocessing Pipeline

- **Scikit-Learn preprocessing pipeline**
  - Systematic and efficient way to streamline and automate the data preprocessing
  - Ensures consistent application of preprocessing steps to train, validation, and test data
  - Enhances reproducibility, readability, and ease of deployment

- **Pipeline functionalities**
  - Filling in missing values
  - Scaling the data to mitigate skewness
  - Standardizing features by making the mean to 0 and scaling to unit variance
  - Encoding string categorical features into numerical representation
  - Dropping features with low correlations
  - Feature engineering

Model Building and Evaluation

Model Building and Evaluation

- Training and evaluating model is as simple as passing in the prepared dataset into few Scikit-Learn APIs
- Scikit-Learn APIs are very consistent across various machine learning algorithms and evaluation metrics
- Assess for underfitting and overfitting by analyzing the model's performance on training and validation datasets
- Integrate the preprocess pipeline with the final model to enable us to export the pipeline for predicting on test data