

---

# Project 4: Regression Analysis and Define Your Own Task!

---

Due on March 16, 2022, 11:59 pm

## 1 Introduction

Regression analysis is a statistical procedure for estimating the relationship between a target variable and a set of features that jointly inform about the target. In this project, we explore specific-to-regression feature engineering methods and model selection that jointly improve the performance of regression. You will conduct different experiments and identify the relative significance of the different options.

## 2 Datasets

You should take steps in section 3 on the following datasets.

### 2.1 Dataset 1: Diamond Characteristics

Valentine's day might be over, but we are still interested in building a bot to predict the price and characteristics of diamonds. A synthetic diamonds dataset can be downloaded from this [link](#). This dataset contains information about 53,940 round-cut diamonds. There are 10 variables (*features*) and for each sample, these features specify the various properties of the sample. Below we describe these features:

- **carat**: weight of the diamond (0.2–5.01);
- **cut**: quality of the cut (Fair, Good, Very Good, Premium, Ideal);
- **color**: diamond colour, from J (worst) to D (best);
- **clarity**: a measurement of how clear the diamond is (I1 (worst), SI2, SI1, VS2, VS1, VVS2, VVS1, IF (best));
- **x**: length in mm (0–10.74)
- **y**: width in mm (0–58.9)
- **z**: depth in mm (0–31.8)
- **depth**: total depth percentage =  $z / \text{mean}(x, y) = 2 * z / (x + y)$  (43–79)
- **table**: width of top of diamond relative to widest point (43–95)

In addition to these features, there is the target variable: i.e what we would like to predict:

- **price**: price in US dollars (\$326–\$18,823);

## 2.2 Dataset 2: Gas Turbine CO and NOx Emission Data Set

Being able to predict the greenhouse gas emissions in a particular region may be more important than assessing the price of diamonds. This dataset can be downloaded from this [link](#). The dataset contains 36733 instances of 11 sensor measurements aggregated over one hour (by means of average or sum) from a gas turbine located in Turkey's north western region for the purpose of studying flue gas emissions, namely CO and NOx (NO + NO2).

**There are 5 CSV files for each year. Concatenate all data points and add a column for the corresponding year and treat it as a categorical feature.**

There are two types of gas studied in this project:

- NOx
- CO

*Pick one by your choice and drop the other. Important: Do not use any of them as a feature.*

## 3 Required Steps

In this section, we describe the setup you need to follow. Follow these steps to process *both* datasets in Section 2.

### 3.1 Before Training

Before training an algorithm, it's always essential to inspect the data. This provides intuition about the quality and quantity of the data and suggests ideas to extract features for downstream ML applications. In this following section we will address these steps.

#### 3.1.1 Handling Categorical Features

A categorical feature is a feature that can take on one of a limited number of possible values. A preprocessing step is to convert categorical variables into numbers and thus prepared for training.

One method for numerical encoding of categorical features is to assign a scalar. For instance, if we have a "Quality" feature with values {Poor, Fair, Typical, Good, Excellent} we might replace them with numbers 1 through 5. If there is no numerical meaning behind categorical features (e.g. {Cat, Dog}) one has to perform "one-hot encoding" instead.

#### 3.1.2 Standardization

Standardization of datasets is a common requirement for many machine learning estimators; they might behave badly if the individual features do not more-or-less look like standard normally distributed data: Gaussian with zero mean and unit variance. If a feature has a variance that is orders of magnitude larger than others, it might dominate the objective function and make the estimator unable to learn from other features correctly as expected.

Standardize feature columns and prepare them for training. **(Question 1)**

### 3.1.3 Data Inspection

The first step for data analysis is to take a close look at the dataset<sup>1</sup>.

- Plot a heatmap of the Pearson correlation matrix of the dataset columns. Report which features have the highest absolute correlation with the target variable. **In the context of each dataset, describe what this high correlation suggests.** (Question 2)
- Plot the histogram of numerical features. What preprocessing can be done if the distribution of a feature has high skewness? (Question 3)
- Construct and inspect the box plot of categorical features vs target variable. What do you find? (Question 4)
- For the Diamonds dataset, plot the counts by color, cut and clarity. (Question 5)
- For the Gas Emission dataset, plot the yearly trends for each feature and compare them. The data points don't have timestamps but you may assume the indices are times. (Question 6)

### 3.1.4 Feature Selection

- `sklearn.feature_selection.mutual_info_regression` function returns estimated mutual information between each feature and the label. Mutual information (MI) between two random variables is a non-negative value which measures the dependency between the variables. It is equal to zero if and only if two random variables are independent, and higher values mean higher dependency.
- `sklearn.feature_selection.f_regression` function provides F scores, which is a way of comparing the significance of the improvement of a model, with respect to the addition of new variables.

You may use these functions to select most important features. How does this step affect the performance of your models in terms of test RMSE? Briefly describe your reasoning. (Question 7)

## 3.2 Training

Once the data is prepared, we would like to train multiple algorithms and compare their performance using average RMSE from 10-fold cross-validation (please refer to part 3.3).

### 3.2.1 Linear Regression

What is the objective function? Train ordinary least squares (linear regression without regularization), as well as Lasso and Ridge regression, and compare their performances. Answer the following questions.

- Explain how each regularization scheme affects the learned hypotheses. (Question 8)
- Report your choice of the best regularization scheme along with the optimal penalty parameter and briefly explain how it can be computed. (Question 9)

---

<sup>1</sup>For exploratory data analysis, one can try [pandas-profiling](#)

- Does feature scaling play any role (in the cases with and without regularization)? Justify your answer. (Question 10)
- Some linear regression packages return  $p$ -values for different features<sup>2</sup>. What is the meaning of them and how can you infer the most significant features? (Question 11)

### 3.2.2 Polynomial Regression

Perform polynomial regression by crafting products of raw features up to a certain degree and applying linear regression on the compound features. You can use `scikit-learn` library to build such features. Avoid overfitting by proper regularization. Answer the following:

- Look up for the most salient features and interpret them. (Question 12)
- What degree of polynomial is best? What does a very high-order polynomial imply about the fit on the training data? How do you choose this parameter? (Question 13)
- For the diamond dataset it might make sense to craft new features such as  $z = x_1 \times x_2$ , etc. Explain why this might make sense and check if doing so will boost accuracy. (Question 14)

### 3.2.3 Neural Network

Try a multi-layer perceptron (fully connected neural network). You can simply use `sklearn` implementation and compare the performance. Then answer the following:

- Why does it do much better than linear regression? (Question 15)
- Adjust your network size (number of hidden neurons and depth), and weight decay as regularization. Find a good hyper-parameter set systematically. (Question 16)
- What activation function should be used for the output? You may use none. (Question 17)
- What is the risk of increasing the depth of the network too far? (Question 18)

### 3.2.4 Random Forest

Apply a random forest regression model on datasets, and answer the following.

- Random forests have the following hyper-parameters:
  - Maximum number of features;
  - Number of trees;
  - Depth of each tree;

Fine-tune your model. Explain how these hyper-parameters affect the overall performance? Do some of them have regularization effect? (Question 19)

- Why does random forest perform well? (Question 20)

---

<sup>2</sup>E.g: `scipy.stats.linregress` and `statsmodels.regression.linear_model.OLS`

- Randomly pick a tree in your random forest model (with maximum depth of 4) and plot its structure. Which feature is selected for branching at the root node? What can you infer about the importance of features? Do the important features match what you got in part 3.2.1? (Question 21)

### 3.2.5 LightGBM, CatBoost and Bayesian Optimization

Boosted tree methods have shown advantages when dealing with tabular data, and recent advances make these algorithms scalable to large scale data and enable natural treatment of (high-cardinality) categorical features. Two of the most successful examples are [LightGBM](#) and [CatBoost](#).

Both algorithms have many hyperparameters that influence their performance. This results in large search space of hyperparameters, making the tuning of the hyperparameters hard with naive random search and grid search. Therefore, one may want to utilize “smarter” hyperparameter search schemes. We specifically explore one of them: Bayesian optimization.

In this part, **pick either one** of the datasets and apply LightGBM and CatBoost. If you do both, we will only look at the first one.

- Read the documentation of LightGBM and CatBoost and experiment on the picked dataset to determine the important hyperparameters along with a proper search space for the tuning of these parameters. (Question 22)
- Apply Bayesian optimization using `skopt.BayesSearchCV` from `scikit-optimize` to search good hyperparameter combinations in your search space. Report the best hyperparameter found and the corresponding RMSE, for both algorithms. (Question 23)
- Interpret the effect of the hyperparameters using the Bayesian optimization results: Which of them helps with performance? Which helps with regularization (shrinks the generalization gap)? Which affects the fitting efficiency? Endorse your interpretation with numbers and visualizations. (Question 24)

### 3.3 Evaluation

Perform 10-fold cross-validation and measure average RMSE errors for training and validation sets. Why is the training RMSE different from that of validation set? (Question 25)

For random forest model, measure “Out-of-Bag Error” (OOB) as well. Explain what OOB error and  $R^2$  score means given this [link](#). (Question 26)

---

# Show Us Your Skills: Twitter Data

---

## Introduction

As a culmination of the four projects in this class, we introduce this final dataset that you will explore and your task is to walk us through an end-to-end ML pipeline to accomplish any particular goal: regression, classification, clustering or anything else. **This is a design question and it is going to be about 30% of your grade.**

Below is a description and some small questions about the provided dataset to get you started and familiarized with the dataset:

### 3.4 About the Data

Download the training tweet data<sup>3</sup>. The data consists of 6 text files, each one containing tweet data from one hashtag as indicated in the filenames.

Report the following statistics for each hashtag, i.e. each file (**Question 27**):

- Average number of tweets per hour
- Average number of followers of users posting the tweets per tweet (to make it simple, we average over the number of tweets; if a users posted twice, we count the user and the user's followers twice as well)
- Average number of retweets per tweet

Plot “number of tweets in hour” over time for #SuperBowl and #NFL (a bar plot with 1-hour bins). The tweets are stored in separate files for different hashtags and files are named as `tweet_[#hashtag].txt`. (**Question 28**)

**Note:** The tweet file contains one tweet in each line and tweets are sorted with respect to their posting time. Each tweet is a **JSON** string that you can load in Python as a dictionary. For example, if you parse it to object `json_object = json.loads(json_string)`, you can look up the time a tweet is posted by:

```
json_object['citation_date']
```

You may also assess the number of retweets of a tweet through the following command:

```
json_object['metrics']['citations']['total']
```

Besides, the number of followers of the person tweeting can be retrieved via:

```
json_object['author']['followers']
```

The time information in the data file is in the form of UNIX time, which “encodes a point in time as a scalar real number which represents the number of seconds that have passed since the beginning of 00:00:00 UTC Thursday, 1 January 1970” (see [Wikipedia](#) for details). In Python, you can convert it to human-readable date by

```
import datetime
datetime_object = datetime.datetime.fromtimestamp(unix_time)
```

The conversion above gives out a `datetime` object storing the date and time in your local time zone corresponding to that UNIX time.

---

<sup>3</sup><https://ucla.box.com/s/24oxnhsoj6kpxhl6gyvuck25i3s4426d>

In later parts of the project, you may need to use the PST time zone to interpret the UNIX timestamps. To specify the time zone you would like to use, refer to the example below:

```
import pytz
pst_tz = pytz.timezone('America/Los_Angeles')
datetime_object_in_pst_timezone =
↪ datetime.datetime.fromtimestamp(unix_time, pst_tz)
```

For more details about `datetime` operation and time zones, see <https://medium.com/@eleroy/10-things-you-need-to-know-about-date-and-time-in-python>. Follow the steps outlined below: (Question 29)

- Describe your task.
- Explore the data and any metadata (you can even incorporate additional datasets if you choose).
- Describe the feature engineering process. Implement it with reason: Why are you extracting features this way - why not in any other way?
- Generate baselines for your final ML model.
- A thorough evaluation is necessary.
- Be creative in your task design - use things you have learned in other classes too if you are excited about them!

We value creativity in this part of the project, and your score is partially based on how unique your task is. Here are a few pitfalls you should avoid (there are more than this list suggests):

- **DO NOT** perform shoddy sentiment analysis on Tweets: running a pre-trained sentiment analysis model on each tweet and correlating that sentiment to the score in time would give you an obvious result.
- **DO NOT**. *only* include trivial baselines: In sentiment analysis, for example, if you are going to try and train a Neural Network or use a pre-trained model, your baselines need to be competitive. Try to include alternate network architectures in addition to simple baselines such as random or naive Bayesian baselines.

Here we list a few project directions that you can consider and modify. These are not complete specifications. You are free and are encouraged to create your projects /project parts (that may get some points for creativity). The projects you come up with should match or exceed the complexity of the following 3 suggested options:

- **Time-Series Correlation between Scores and Tweets:** Since this tweet dataset contains tweets that were posted before, during, and after the Superbowl, you can find time-series data that have the real-time score of the football game as the tweets are being generated. This score can be used as a dynamic label for your raw tweet dataset: there is an alignment between the tweets and the score. You can then train a model to predict, given a tweet, the team that is winning. Given the score change, can you generate a tweet using an ensemble of sentences from the original data (or using a generative model that is more sophisticated)?

---

(15:00 - 1st) S.Hauschka kicks 70 yards from SEA 35 to NE -5. D.Amendola to NE 18 for 23 yards (D.Shead).

---

**1st & 10 at NE 18**

(14:57 - 1st) (Shotgun) T.Brady pass short right to R.Gronkowski to NE 20 for 2 yards (B.Irvin).

---

**2nd & 8 at NE 20**

(14:15 - 1st) T.Brady pass short left to D.Amendola pushed ob at NE 26 for 6 yards (B.Maxwell).

---

**3rd & 2 at NE 26**

(13:47 - 1st) (Shotgun) T.Brady pass short left to S.Vereen to NE 31 for 5 yards (K.Wright) [M.Bennett].

---

Figure 1: A sample of the significant events in the game that you can easily find on the internet. [Here](#) is one link that has the time-indexed events.

- **Character-centric time-series tracking and prediction:** In the #gopatrics dataset, there are several thousand tweets mentioning “Tom Brady” and his immediate success/failure during the game. He threw 4 touchdowns and 2 interceptions, so fan emotions about Brady throughout the game are fickle. Can we track the average perceived emotion across tweets about each player in the game across time in each fan base? Note that this option would require you to explore ways to find the sentiment associated with each player in time, not to an entire tweet. Can we correlate these emotions with the score and significant events (such as interceptions or fumbles)? Using these features, can you predict the MVP of the game? Who was the most successful receiver? The MVP was Brady.
- **Library of Prediction Tasks given a tweet:** Predict the hashtags or how likely it is that a tweet belongs to a specific team fan. Predict the number of retweets/likes/quotes. Predict the relative time at which a tweet was posted.

## Submission

Your submission should be made to both of the two places: BruinLearn and Gradescope within BruinLearn.

**BruinLearn** Please submit a zip file containing your **report**, and your **codes** with a **readme file** on how to run your code to CCLE. The zip file should be named as “Project1\_UID1\_UID2....\_UIDn.zip”

where UIDx’s are student ID numbers of the team members. **Only one submission per team is required.** If you have any questions, please ask on Piazza or through email. Piazza is preferred.

**Gradescope** Please submit your report to Gradescope as well. Please specify your group members in Gradescope. It is very important that you assign each part of your report to the question number provided in the Gradescope template.