Large-Scale Data Mining: Models and Algorithms **ECE 219 Winter 2021**
Prof. Vwani Roychowdhury UCLA, Department of ECE

Project 4: Regression Analysis

Due on March 19, 2021, 11:59 pm

1 Introduction

Regression analysis is a statistical procedure for estimating the relationship between a target variable and a set of describing features. In this project, we explore common practices for best performance of regression. You will conduct different experiments and identify the significance of practices that we suggest below.

As for your report, whenever you encounter several options, explain what choices are better to make. You need to justify your claim and back it with numerical results. You are not asked to experiment all combinations of different choices in a tedious way. Instead, in some cases you may just report how an observations lead you to make a particular choice and move forward with that choice fixed. Casually speaking, you may at times make greedy decisions over the tree of your choices!

Answer all questions very concisely and to-the-point. Most questions require short answers with few keywords.

2 Datasets

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

2.1 Bike Sharing Dataset

This dataset can be downloaded from this **link**. Bike sharing dataset provides count number of rental bikes based on some timing and environmental conditions. You can find feature descriptions in the provided link. We are asking you to perform all your analyses based on three different target variables:

- casual: count of casual users
- registered: count of registered users
- cnt: count of total rental bikes including both casual and registered

Perform data inspection in section 3.1.1 for all three targets and continue the project with total count (third item). As you notice, there are two files, `hour.csv` and `day.csv`. Use `day.csv` for this project.

2.2 Suicide Rates Overview 1985 to 2016

This dataset can be downloaded from **this link.** This compiled dataset pulled from four other datasets linked by time and place, and was built to find signals correlated to increased suicide rates among different cohorts globally, across the socioeconomic spectrum. We are asking you to perform Regression analysis on about 27820 datapoints using these variables:

- country
- year
- sex
- age
- population
- gdp_for_year ($)
- gdp_per_capita ($)
- generation

The target variables are:

- suicides_no
- suicides/100k pop

There exist only one file in the provided link. Perform data inspection in section 3.1.1 for both targets and continue the project with "suicides/100k pop".

2.3 Video Transcoding Time Dataset

Video transcoding time dataset can be downloaded from this link. It includes input and output video characteristics along with their time taken for different valid transcodings. The dataset has 68784 data points in total, and each instance has 19 features as described below:

- duration = duration of video in second;
- codec = coding standard used for the input video (e.g. `flv`, `h264`, `mpeg4`, `vp8`);
- height, width = height and width of video in pixels;
- bitrate = bits that are conveyed or processed per unit of time for input video;
- frame rate = input video frame rate(fps);
- i = number of i frames in the video, where i frames are the least compressible but don't require other video frames to decode;
- p = number of p frames in the video, where p frames can use data from previous frames to decompress and are more compressible than I-frames;
- b = number of b frames in the video, where b frames can use both previous and forward frames for data reference to get the highest amount of data compression;
- frames = number of frames in video;
- i-size, p-size, b-size = total size in byte of i, p, b frames;
- size = total size of video in byte;
- o-codec = output codec used for transcoding;

- o-bitrate = output bitrate used for transcoding;
- o-framerate = output framerate used for transcoding;
- o-width, o-height = output width and height in pixel used for transcoding.

There are two files in the downloaded folder. Only use `transcoding-mesurment.tsv` for your project. Please notice that this file contains 19 features above as well as following two attributes:

- umem = total codec allocated memory for transcoding;
- utime = total transcoding time for transcoding.

Note that the target variable is transcoding time, which is the last attribute "utime" in the data file.

3 Required Steps

In this section, we describe the setup you need to follow. Take these steps on the datasets in section 2. (Take whichever steps that may apply to each dataset.).

3.1 Before Training

Before training an algorithm, it's always essential to inspect data and understand how it looks like. Also, raw data might need some preprocessing. In this section we will address these steps.

3.1.1 Data Inspection

The first step for data analysis is to take a close look at the dataset[1].

- Plot a heatmap of Pearson correlation matrix of dataset columns. Report which features have the highest absolute correlation with the target variable and what that implies. **(Question 1)**
- Plot the histogram of numerical features. What preprocessing can be done if the distribution of a feature has high skewness? **(Question 2)**
- Inspect box plot of categorical features vs target variable. What intuition do you get? **(Question 3)**
- For bike sharing dataset, plot the count number per day for a few months. Can you identify any repeating patterns in every month? **(Question 4)**
- For the suicide rate dataset, pick the top 10 countries that have the longest time-span of records (in terms of years). Plot the suicide rate ("`suicides/100k pop`") against time for different age groups and gender, and explain your observations. (Hint: try `seaborn.relplot`) **(Question 5)**
- For video transcoding time dataset, plot the distribution of video transcoding times, what can you observe? Report mean and median transcoding times. **(Question 6)**

[1]For exploratory data analysis, one can try pandas-profiling

3.1.2 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.

For some other cases, e.g. when encoding time stamps such as {`Mon, ..., Sun`} or {`Jan, ..., Dec`} it might make sense to perform either one. In those cases the learning algorithm of choice and numerical results can lead our way. Can you explain a trade-off here? (Hint: let us assume we perform linear regression, what information does one-hot encoding discard, and what assumption should hold strongly if we perform the scalar encoding instead?) **(Question 7)**
For the suicide rate dataset, the number of unique countries for the variable "country" is pretty high. We suggest you to group these countries into same continent countries such as Europe, North America, South America, Middle East and Asia. Using a new variable with lower number of classes in this category, you may proceed with one-hot encoding.

3.1.3 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 8)**

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? **(Question 9)**

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 10)**
- Report your choice of the best regularization scheme along with the optimal penalty parameter and briefly explain how it can be computed. **(Question 11)**
- Does feature scaling play any role (in the cases with and without regularization)? Justify your answer. **(Question 12)**
- Some linear regression packages return p-values for different features[2]. What is the meaning of them and how can you infer the most significant features? **(Question 13)**

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 14)**
- What degree of polynomial is best? What reasons would stop us from too much increase of the polynomial degree? How do you choose that? **(Question 15)**
- For the transcoding dataset it might make sense to craft inverse of certain features such that you get features such as $\frac{x_i x_j}{x_k}$, etc. Explain why this might make sense and check if doing so will boost accuracy. **(Question 16)**

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 17)**
- Adjust your network size (number of hidden neurons and depth), and weight decay as regularization. Find a good hyper-parameter set systematically. **(Question 18)**
- What activation function should be used for the output? You may use none. **(Question 19)**
- What reasons would stop us from too much increase of the depth of the network? **(Question 20)**

[2]E.g: `scipy.stats.linregress` and `statsmodels.regression.linear_model.OLS`

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 21)**

- Why does random forest perform well? **(Question 22)**

- 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 23)**

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 one of the datasets and apply LightGBM and CatBoost.

- Read the documentation of LightGBM and CatBoost and experiment on the picked dataset to determined the important hyperparameters along with a proper search space for the tuning of these paarmeters. **(Question 24)**

- Apply Bayesian optimization using `skopt.BayesSearchCV` from `scikit-optmize` to search good hyperparameter combinations in your search space. Report the best hyperparameter found and the corresponding RMSE, for both algorithms. **(Question 25)**

- 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 26)**

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 27)**

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 28)**

4 Submission

Please submit your report to Gradescope with a link provided on CCLE. Please submit your work only by one of the team members.