Final Project – Basic Probability: Programming

Instructors: Lina Murady

December 7, 2020

1 Project Description

In this project you will perform your own data analysis. Your task is to use a technique known as linear regression (see below) to predict the median price of houses in suburbs of Boston in the 1980s. The relevant information about the data can be found here. The variable that you need to predict is stored in the last column, called MEDV.[1]

This project will teach you two things: first, it introduces you to a continuous distribution, namely the Gaussian. Linear regression and other Gaussian models are very useful in practice and you will see them a lot when working with data. Second, you will have to interpret your results. Rather than just running the algorithm, you will have to make sense of the algorithm's output.

2 Data

The data is present in the package `scikit-learn`. Install the package.[2] After that, run:

```
# Import the datasets
from sklearn import datasets

# Load dataset that we will use
boston = datasets.load_boston()

# Create features variable matrix
features = boston.data

# Collect the target variable (MEDV)
target = boston.target
```

[1] Assignment based on original material by Jakub Dotlačil.

[2] Note when working in Colab this is already installed

In the features matrix, each row is one case. That is, each row represents information about one house. Each case has 13 features. The interpretation of each feature can be accessed by boston.DESCR. The values to be predicted (the median price of each house) are stored per case in target. For each case, you should try to combine the features you choose as predictors of the corresponding value in the target.

3 Linear Regression

To familiarise yourself with linear regression and its implementation, watch the videos of Sections 2.1–4.5 (this is quite a lot of movies, but most of them just review linear algebra and can be skipped) of Andrew Ng's machine learning course. You can also sign up for it on coursera to see them in higher resolution. Here, we will only shortly highlight the math of linear regression.

From a probabilistic view point, we assume that our data are independently distributed. This is a weaker assumption than we have made so far because we do not assume that the distributions of our data points are identical! In fact, linear regression is all about working with a different distribution for each data point.

In linear regression we assume that each data point x_i, $(1 \leq i \leq n)$ follows a Gaussian or normal distribution. The probability density function of this distribution is

$$P(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right) . \tag{1}$$

What is special about the normal distribution is that its mean and variance are equal to its parameters. In particular, we have $\mathbb{E}[X] = \mu$ and $\text{var}(X) = \sigma^2$ for $X \sim \mathcal{N}(\mu, \sigma^2)$.

Linear regression assumes that the variance parameter σ^2 is shared by all distributions but that each data point x_i was generated from a Gaussian with mean μ_i. This mean is computed as $\mu_i = w^\top \vec{x_i}$ where w is a weight vector and $\vec{x_i}$ is a vector of predictors[3] of x_i. The mean is thus a linear combination of the data point's predictors. Each coefficient in w relates its predictor linearly to the output variable.

4 Your Task

Your task is to implement linear regression for the Boston housing data set. You first implement a baseline regression model. A baseline of this study is the most basic implementation of the linear regression method on this dataset (you have

[3] For the sake of explanation, we make a strict difference between features and predictors. Features are attributes of a data point whereas predictors are values supplied to the model.

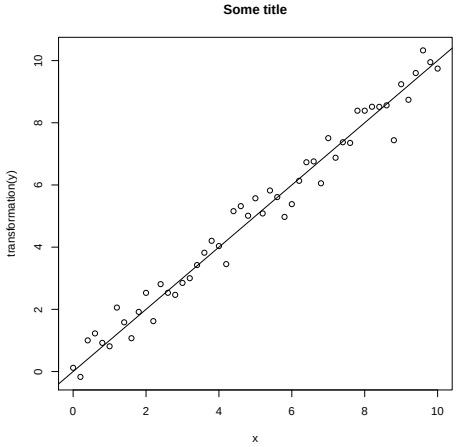

Figure 1: Plot showing relationship between two variables

to justify your choice of baseline!). Thereafter, you try to improve your model and get better predictions.

You have to submit a code that (i) runs linear regression on the dataset, (ii) specifies the baseline model, (iii) specifies the best model, and prints R^2 (see below) for both baseline and best model, (iv) plots a graph showing a relation between one variable (of your choosing) on the x-axis, and the predicted value on the y-axis.

5 Evaluating Linear Regression

Since linear regression makes continuous predictions, it cannot simply be evaluated on a wrong/correct basis. The error of linear regression is induced by the distances of the predictions to the true values. These distances are known as residuals. The sum of their squares is the (squared) error[4].

Take a look at the plot in Fig. 1. If the line was the output of a linear regression model, the residuals would be the vertical distances of the points to the line.

Since the residual error grows as a function of the number of data points, it is not a meaningful way of evaluating a regression model. However, the residuals are again normally distributed. Standardly, it is assumed that $\mu = 0$ for that distribution.[5] If the variance of their distribution is small, the regression line provides a tight approximation to the true values. If the variance of that distribution is large, the approximation is pretty shabby.

[4] Squaring is necessary here because the negative and positive residuals would cancel each other out otherwise.

[5] This assumption is made because residuals can be both positive and negative and there is no reason to assume that residuals should be "more positive" or "more negative" on average.

The standard quantity for assessing linear regression models is the R^2 which can be computed as

$$R^2 = 1 - \frac{\sum_{i=1}^{n} \left(x_i - w^\top h(x_i)\right)^2}{\sum_{i=1}^{n} \left(x_i - \frac{1}{n}\sum_{j=1}^{n} x_j\right)^2} . \tag{2}$$

Notice that the numerator is the sum of the residual errors and the denominator is the sample variance of the data. The R^2 thus expresses what fraction of the observed variance in the data is captured by the model. The R^2 has the convenient property of being standardized, i.e. to be bounded by 0 and 1. The better the linear regression fits the data, the closer the value of R^2 is to 1.

6 Types of Features and interpretation

Roughly speaking there are two types of features: continuous ones and categorical ones.

Continuous Features Let us assume that our model only has one continuous feature f with coefficient α. This means that for each point increase in f, the predicted value experiences a change of α. If α was -2, the predicted value would decrease by two for each point increase of f. This is a general property of continuous features in linear regression: their values are linearly related to the output value.

Categorical Features Categorical features are all features for which the assumption that their values are linearly related to the output does not make sense. Consider a feature *gender*. First of all, it needs to be turned into a number to be useful for regression. So let us define the mapping male $\mapsto 0$, female $\mapsto 1$.[6] Then we would certainly not want to assume that the values of *gender* are linearly related to whatever output we are dealing with. Rather, it is the level[7] *female* that causes a change in the output.

Whenever we are dealing with categorical features, we need to define a standard or default level. This level can be chosen arbitrarily (in the example above it was chosen to be male). All coefficients of the other levels then have to be interpreted with respect to that default. To make this concrete, let us assume that we model football-playing ability and that our categorical features is *nationality* with levels Holland, Germany and Switzerland. We arbitrarily chose Holland as default. We then introduce two(!) predictors into the model, namely Germany and Switzerland. Each of these predictors is either 1 (when the nationality matches the level) or 0 (otherwise). After training our model, we get a coefficient of 5 for Germany and -3 for Switzerland. The way to interpret these coefficients is that being

[6] This mapping is arbitrary in that we might also map males to 1 and females to 0.

[7] The values of categorical features are often called *levels*.

German increases your ability in football by 5 compared to being Dutch whereas being Swiss decreases it by 3 (again compared to being Dutch).

The reason that we need to introduce 3 levels when dealing with 2 predictors is that if we assigned the numbers 1 and 2 to Germany and Switzerland, we would again assume a linear relationship based on nationality. In general, introducing a predictor for each non-standard level allows us to estimate a separate coefficient per level.

A pre-final remark: it is not always clear whether a feature should be seen as continuous or categorical. In the housing data, we have the RAD feature. Should that be continuous or categorical? This is something you may want to experiment with.

7 Learning

Learning a linear regression model can be done with the technique of least squares. This basically means that you try to iteratively minimise the sum of squares of the residuals. The details are explained in the videos.

8 Improving your Algorithm

There are several ways to improve your algorithm. As discussed in the videos, you may use regularisation. You can also try to build new predictors by transforming or combining features. All of this is up to you. The important thing is that in the end you can show a real improvement through an increase in R^2.

9 Deliverables

- **Code:** A code that you have written, collected in one Jupyter notebook file.
- The code can use libraries (numpy, matplotlib), but *it must not use a previously implemented linear regression*! You have to build the regression yourself.
- A notebook embeds code, documentation, and results in one file. It's the best alternative to a *report* in a programming course. Use Markdown cells for documentation: summarise what you picked as the baseline model, what you picked as the final model, and why you made this choice.

10 Assessment

The lecturer of the course determines the final grade for the project.

Here is the break-down of what I consider:

- 1 points: The code should be well documented, so that I can understand what each class and method and function does.
- 2 points: The code should run (from Jupyter notebook) and (at least) print the baseline model and its R^2, the best model and its R^2, and plot at least one graph.
- 3 points: Linear regression is implemented correctly and it works with numpy arrays (e.g., it does not loop through lists).
- 1 point: Plotting function works and it plots a graph showing the relation between (some subset of) features and the target variable. The data points are printed as dots and the linear regression fit is also shown in the graph.
- 1 point: The best model has R^2 higher than 0.7
- 1 point: The best model has R^2 higher than 0.75
- 1 point: The best model has R^2 higher than 0.8