In the Name of God

Statistical Pattern Recognition

Home Work I

Linear and Logistic Regression

Assignment Date: **19 Mehr** Submission Deadline: **3 Aban**

Contents

1 Objectives and Precautions

In this homework you will learn:

- How to understand a dataset (so called EDA: Explanatory Data Analysis)
- How to implement a linear and logistic regression model
- How to validate your models

Keep in mind that you can only use these three python libraries in your implementations:

- Pandas (to read and manipulate your datasets)
- Numpy (to work with arrays and matrices)
- Matplotlib.pyplot (to plot your samples and regression lines)

Also note that:

- You have 2 weeks to complete this homework.
- The home work instructions and datasets will be shared with you in your **Quera class**.
- Save your results and answers in any format you want, this will be your report.
- Save your code files and your report as a zip file and upload in Quera. (naming format is "your names.zip" for example "Achraf_Hakimi_Mohamed_Salah.zip")
- Late submission strategy is: 70 percent score for one day delay and 50 percent for 2 days delay. Submissions after 2 days will not be graded.
- Use only the python programming language.
- Feel free to ask your questions in the telegram channel.
- Do not copy other works, write your own code.

Thank you. Good Luck.

2 Linear Regression

Linear regression is a linear model, that assumes a linear relationship between the input variables (x) and the single output variable (y). More specifically, that y can be calculated from a linear combination of the input variables (x).

2.1 Understanding the data: climate_change.csv

There have been many studies documenting that the average global temperature has been increasing over the last century. The consequences of a continued rise in global temperature will be dire. Rising sea levels and an increased frequency of extreme weather events will affect billions of people. In this problem, we will attempt to study the relationship between average global temperature and several other factors.

The file climate_change.csv contains climate data from May 1983 to December 2008. The available variables include:

- Year: the observation year.
- Month: the observation month.
- Temp: the difference in degrees Celsius between the average global temperature in that period and a reference value. This data comes from the Climatic Research Unit at the University of East Anglia. (In Fig. 1 You can see how average temperature changes over the years.)
- CO2, N2O, CH4, CFC.11, CFC.12: atmospheric concentrations of carbon dioxide (CO2), nitrous oxide (N2O), methane (CH4), trichlorofluoromethane (CCl3F; commonly referred to as CFC-11) and dichlorodifluoromethane (CCl2F2; commonly referred to as CFC-12), respectively.
- Aerosols: the mean stratospheric aerosol optical depth at 550 nm. This variable is linked to volcanoes, as volcanic eruptions result in new particles being added to the atmosphere, which affect how much of the sun's energy is reflected back into space. This data is from the Godard Institute for Space Studies at NASA.
- TSI: the total solar irradiance (TSI) in W/m2 (the rate at which the sun's energy is deposited per unit area). Due to sunspots and other solar phenomena, the amount of energy that is given off by the sun varies substantially with time. This data is from the SOLARIS-HEPPA project website.
- MEI: multivariate El Nino Southern Oscillation index (MEI), a measure of the strength of the El Nino/La Nina-Southern Oscillation (a weather effect in the Pacific Ocean that affects global temperatures). This data comes from the ESRL/NOAA Physical Sciences Division.

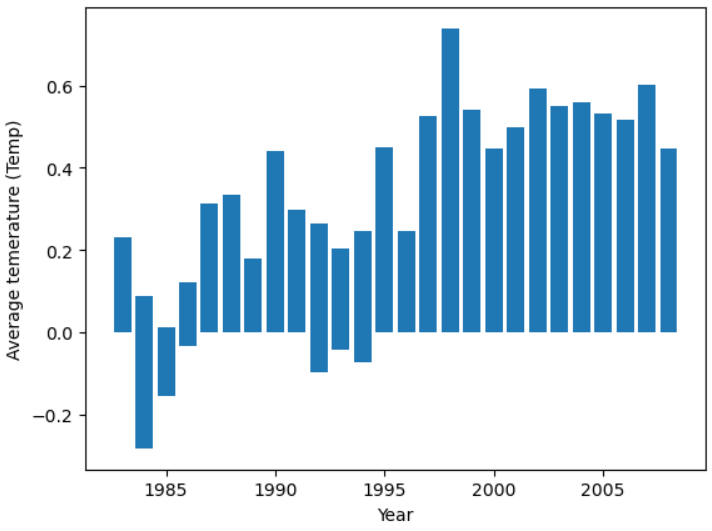

Figure 1: How average temperature changes over the years. This bar plot was generated using the command: plt.bar(data['Year'],data['Temp'])

Here we are interested in prediction of the average temperature based on some of the above features using linear regression. Follow these instructions to understand and prepare your dataset for the next section:

1. Load the dataset (you can use: pd.read_csv('climate_change.csv'))
2. Take a look at some samples of your dataset using (you can use: data.sample(5))
3. How many sample are there in the dataset? (you can use: data.shape)
4. Average temperature is under the heading 'Temp'. What is the mean value of Temp? How about the mean values of CO2, NO2 and CH4? (you can use: data.describe())
5. How spread out are the Temp values from the mean of Temp? (this measure is called standard deviation (std))
6. Which variables have more linear relationship with the Temp variable? (this measure is called correlation) ((you can use: data.corr()) Also define correlation.
7. Plot data samples with CO2 values on the x axis and Temp on the y axis. Does temperature increase as CO2 increases? (you can use: plt.scatter(data.CO2,data.Temp))

So we realized which variables have more linear correlation with temperature and saw a scatter plot of how one increases as the other is increased.

2.2 Creating our model: simple linear regression

In order to create your regression model, follow these instructions:

1. Split your data into training set and test set. Pick samples with Year<= 2003 as training set and samples with years> 2003 as test set. This way you will train your regression model on the data between 1983 and 2003, then use this model to predict temperature for the upcoming years 2004 up to 2008. (you can use: training_set = data[data.Year<= 2003])
2. How many training and test samples do you have?
3. Simple regression is when you have only one dependent and one independent variable. So pick only CO2 and Temp variables from both your training and test sets, and drop the rest of the columns. (you can use: training_set=training_set[['Temp','CO2']]
4. Now separate your training and test sets, into feature X (CO2) and label y (Temp).
   (you can use: X_train,y_train= training_set.CO2.values, training_set.Temp.values)
   Attention: The df.values command turn your pandas DataFrames into numpy arrays for better mathematical behaviour in the following sections.
5. Normalize the CO2 values to have standard deviation = 1 and mean = 0. (Note: use the mean and std of your training set to normalize the test set)

Here, it is time to implement your linear regression model. Follow the instructions:

1. Implement linear regression from scratch using the gradient descent algorithm (batch 'or' stochastic). Train your model on the training set.
2. Plot cost function for enough iteration.
3. Try different learning rates and different number of iterations. What do you infer from this experiment?
4. Report the parameters for your regression line. (theta 0 and theta 1)
5. Predict temperature values for both your training and test sets, then report the value of MSE (Meas Squared Error) between true and predicted values.
6. Report your regression line in the format of "y = ax + b".
7. Plot the training and test sets with different colors along with the regression line you modeled.
8. Finally implement your linear regression model using the closed form solution rather that gradient descent algorithm, then repeat 4 to 7.

3 Logistic Regression

In contrast to linear regression that aims to predict an outcome, logistic regression aims to solve binary classification problems. It models the probability of being in one class or another.

3.1 Understanding the data: synthetic_software_defect.csv

Software Defect dataset was made publicly available in December 2004 by NASA in order to encourage data scientists to build predictive models useful in software engineering. Yet your dataset for this problem is synthetically generated from the Software Defect dataset and made public by **Kaggle**.

The file synthetic_software_defect.csv contains 21 feature of a software and a label that shows if the software is defective or not. The available variables include (some of these are software technical words so you do not have to understand their meaning for this homework, only take a look):

- loc: line count of code
- v(g): cyclomatic complexity
- ev(g): essential complexity
- iv(g): design complexity
- n: total operators + operands
- v: volume
- l: program length
- d: difficulty
- i: intelligence
- e: effort
- b: (not provided by data donors)
- t: time estimator
- lOCode: line count
- lOComment: count of lines of comments
- lOBlank: count of blank lines
- lOCodeAndComment: count of lines of code and comments
- uniq_Op: unique operators
- uniq_Opnd: unique operands
- total_Op: total operators
- total_Opnd: total operands
- branchCount: of the flow graph
- defects: (True or False) module has or has not reported any defects

Here we see that our samples are software which belong to two classes: defective or healthy. And we want to build a model to help us classifying a software based on its features. So our problem is binary classification. Follow these instructions to understand and prepare your dataset for the next section:

1. Load the dataset. (you can use: pd.read_csv('synthetic_software_defect.csv', index_col='id'))
2. Take a look at some samples of your dataset. How many sample are there? How many of your features are numerical, categorical, and binary? (you can use: data.info())

3. Name two features that have more correlation with the target column? (you can use: data.corr())

4. Keep those two features you found in the previous question along with the target column, then drop the rest of the features of your dataset. (you can use: data = data[['feature 1', 'feature 2', 'defects']])

5. Are there duplicate samples in your dataset? What percent of your samples are duplicates? Drop the duplicates. (you can use: data.drop_duplicates())

6. Your data has only three columns now and has no duplicates. Plot the histogram of both your selected features. (you can use: plt.hist(data['your feature'], bins=100))

7. What common distribution do these features have? what is the name of this distribution?

8. Your logistic regression model will profit if you transform this data distribution to another one. Use np.log1p() to log-transform your selected features. (you can use: data[['feature 1', 'feature 1']].apply(np.log1p))

9. Whats the difference between np.log1p() and np.log()?

10. Now your data has only 3 columns, no duplicates, and also your features are log-transformed. Plot the histogram of your features again. Compare their distribution before and after the log-trasform.

11. Now split your data into training and test sets. Select rows [0:350] of your preprocessed data as training set and rows [350:400] as test set. (you can use: data_train = data.iloc[0:350])

12. Also divide your training and test sets into features (X) and label (y). (you can use: X_train = data_train[['feature 1','feature 2']] and y_train = data_train['defects'])

13. Now your data may be in 4 parts of X_train, y_train, X_test, y_test. Plot the scatter of your training and test samples with feature 1 on x-axis and feature 2 on y-axis. Use different colors for defective and healthy samples. (you can use: plt.scatter(X_train[feature 1, X_train['feature 2'], color=y_train))

14. Looking at your scatter plot, are defective and healthy samples well separated?

Up to here, you have explored and studied your dataset and prepared training and test sets. It's time to create your regression model. (Keep in mind that it is always a good idea to keep your data code apart from your modeling code.)

3.2 Creating our model: logistic regression

In order to create your model, follow these instructions:

1. Before anything else, if your data type is pandas Dataframe, transform it to numpy. (you can use: X_train = X_train.values) And to check the type of your data you can use: type(X_train).

2. Implement logistic regression from scratch. Train your model on the training set.

3. Plot cost function for enough iteration.

4. Try different learning rates and different number of iterations. What do you infer from this experiment?

5. Report the parameters for your logistic regression line. (theta 0 and theta 1)

6. Classify samples of both your training and test sets, then report the value of accuracy score between true and predicted values.

7. Report your logistic regression line in the format of "y = ax + b".

8. Plot the training and test sets with different colors along with your logistic regression line (this line is also the decision boundary).

4 Bonus

The synthetic software defect dataset you used to train your logistic regression model, is currently under competition in **Kaggle** with the title of "Binary Classification with a Software Defects Dataset".

In your homework you only used a subset of this dataset with only 2 features. The bonus of this homework is to implement binary classification using logistic regression on the complete dataset and participate in this beginner-level competition.

Anyone who submit results to this competition and get a score from its website will get the bonus.You can find the competition homepage **here**.