Artificial Intelligence Term-Project

Linear Regression Explained

Melis Akarçay, MEF University, Istanbul, Turkey

abstract - This term project is about the main methods of machine learning and detailed information about the linear regression.

key words – machine learning, supervised learning, reinforcement learning, regression, linear regression

1. Artificial Intelligence

Artificial intelligence is a branch of computer science. The main aim of AI is to create intelligent machines that react like a human. There are important layers such as recognizing sounds, learning, planning and solving the problem in the process of creating artificial intelligence.

2. Machine Learning

Machine learning aims to enable a machine to produce logical and rational results based on the data given to it. These algorithms develop new behaviors based on data. Thus, machines can do more and adapt to new conditions using the specific features given to them. Simply put, we can say that machines are trained to learn by experiencing how to perform a task. The difference in machine learning is revealed at this point. Machine learning also produces an intelligent product, but it does by making able machines to learn. Machine learning has three types. They are Supervised, Unsupervised and Reinforcement Learning.

2.1. Unsupervised Learning

Unsupervised learning uses a function to estimate an unknown structure over unlabelled data. This method can be explained as a machine trying to learn the algorithm without the teacher. Clustering and dimensionality reduction are techniques for implementing the unsupervised learning.

In order to implement the clustering operation, the similarities between the data must be found and used. Clustering aim is to find the hidden structure of data. The important point is to reduce the similarities between clusters for having an accurate output. The most popular algorithm for clustering is k-means which divide the dataset into k clusters. Dimensionality reduction's aim is to reduce the size of a high-dimensional dataset. The aim is to reduce the features of the dataset. Two main functions of dimensionality reduction are feature selection and feature extraction.

2.2. Reinforcement Learning

Reinforcement learning deals with what actions agents need to take to achieve the highest rewards in an environment. There is an instructor in reinforcement learning, but it does not govern the system as in supervised learning. Instead, when the agent makes a decision, it rewards the system for movements where the decision is correct and penalizes for the wrong. The most well-known reinforcement learning method is Q-learning. Q-learning accumulates the movements and the rewards gained in a table named Q-table. And the agent tries to reach the maximum reward by using the table.

2.3. Supervised Learning

Supervised learning is a method in which the machine learning by using labeled data. If the machine has label data that means the machine has the values of input and output data. The training data consists of both these inputs and outputs. In this learning technique, it generates a function that matches the dataset. Supervised learning approaches are classification and regression. The machine takes input from the dataset then uses learning techniques to classify the objects. In classification, the output is discrete which means categorical. The other approach is regression. In regression, the machine tries to estimate a numerical value from the dataset. It is an analysis method used to measure the relationship between variables.

a. Supervised vs Unsupervised Learning

A function is generated by a machine learning method through training data. With this function, the incoming data tries to be

predicted. The major difference of supervised learning is that the input and output data are contained in the dataset. In Unsupervised Learning, it is the process in which the program try to derive meaning from the raw data without knowing which outputs they produce. Basically, Unsupervised learning finds all kinds of unknown patterns in data.

b. Supervised vs Reinforcement Learning

The main difference is supervised learning is working on existing data but reinforcement learning works on interacting with the environment

3. Linear Regression

Linear regression is obtain the relationship between parameters as a function. Its main purpose is to determine whether there is a linear relationship between the parameters. When we put data into the plot they don't follow a straight line however they follow a linear pattern hence the term linear regression is used. In the implementation, the goal is to estimate a result (Y) for a given parameter (X). One variable considered to be an explanatory (X) variable while the other is considered to be a dependent variable (Y). The dependent variable is the value that we want to explain or forecast. The independent (explanatory) is a variable that explains the other variables. Here is an equation for the relationship between X and Y:

$$Y=\beta_0+\beta_1 X$$

$\beta 0$ and $\beta 1$ are coefficients. For making it simple $\beta 0$ is intercept and $\beta 1$ is the slope. If you increase this variable the intercept moves up or down along the y-axis. β1 is the coefficient for the independent variable and it is able to make a unit change in X then the affects Y for each unit. In high-dimensional datasets, there will be more than one x input and it is called as a hyperplane. The following formula is used for finding the coefficients:

$$\beta_1 = \frac{\sum_{i=1}^{m}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{m}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \bar{x}$$

* m in the formula specifies the size of the sample space.

After getting the coefficients second step is to find predicted values according to the equation. For finding the predicted values all needed is to place X values and getting the output as Y value. Regression line is also created with using the same equation. The difference between the regression line and the observed value is called as the residual. The goal of linear regression is to create a linear model that minimizes the sum of squared residuals.

The terms residual, regression line and data are clearly indicated in Figure 1.

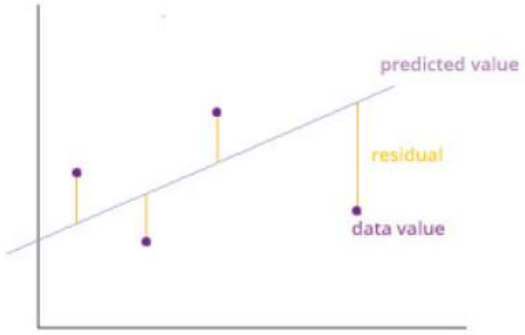

Fig.1. Sample graph

After the model is built, the last part of the regression is to perform performance analysis. If the difference between the predicted value and actual value is not much, that means it can be a good model. There are a few tools for calculating the error rate. Two of these are Root Mean Squared Error and Coefficient of Determination.

a. Coefficient Determination (R^2)

Coefficient determination has three parameters. They are SSR, SSE and SST. SST is called as the total sum of squares. SST formula is shown below:

$$SST = \Sigma (Y - Mean[Y])^2$$

SSE is called as residual/error sum of squares. It tells the difference between the actual and predicted value.

$$SSE = \Sigma (Y - f[Y])^2$$

SSR is the sum of the square of regressions. The calculation is made with summing the square of difference between the predicted value and the mean value.

Coefficient determination formula is :

$$R^2 \equiv 1 - \frac{SS_r}{SS_t}$$

R^2 rate is generally between 0 and 1. If R^2 is close 1 that indicates a good fit. If R^2 is 0 that means there's no correlation between X and Y values.

a. Root Mean Squared Error

RMSE is a measure of the average deviation of the actual from the predicted values.

$$RMSE = \sqrt{\sum_{i=1}^{m} \frac{1}{m}(\hat{y}_i - y_i)^2}$$

2.3.1 Implementation

Linear regression can implement with calculating formulas manually or the user can use the sklearn library. In sklearn library, there is a linearregression() function for creating the model.

4. Basic Example for Linear Regression :

In the paper, I've focused on the basic implementation of linear regression and the implementation with the sklearn library. The first code provided to the paper is taken from M.Jane's blog. The second code is created by R.Goswami. The codes have only two dimensions and properties are head size and brain weight.

4.1. Simple Linear Regression Code

```
# Reading Data
data = pd.read_csv('headbrain.csv')
data.head()

# Collecting X and Y
X = data['Head Size(cm^3)'].values
Y = data['Brain Weight(grams)'].values

# Calculating coefficient

# Mean X and Y
mean_x = np.mean(X)
mean_y = np.mean(Y)

# Total number of values
n = len(X)
```

Fig. 2. Preparation of dataset

Initially, the sample dataset implemented for applying the regression algorithm. In the code, only head size and brain weight features are used. X and Y variables are filled with data under these features. mean_x and mean_y are used to find average head size and brain weight using Numpy library of Python. And n is defined for finding the size of sample space.

```
# Using the formula to calculate b1 and b2
numer = 0
denom = 0

for i in range(n):
    numer += (X[i] - mean_x) * (Y[i] - mean_y)
    denom += (X[i] - mean_x) ** 2
b1 = numer / denom
b0 = mean_y - (b1 * mean_x)

# Printing coefficients
print("Coefficients")
print(b1, b0)
```

Fig. 3. Finding the coefficients

As a second part, numer and denom values are created for calculating B0 and B1 values. Numer represents the sum of (X[i] – mean(X))*(Y[i] – mean(Y)) all state space. Denom represents the $(X[i] – mean(X))^2$. B1 represents slope and B0 represents intercept in the line equation.

```
max_x = np.max(X) + 100
min_x = np.min(X) - 100

# Calculating regression line values x and y
x = np.linspace(min_x, max_x, 1000)
y = b0 + b1 * x

# Ploting Line
plt.plot(x, y, color='#58b970', label='Regression Line')
# Ploting Scatter Points
plt.scatter(X, Y, c='#ef5423', label='Scatter Plot')

plt.xlabel('Head Size in cm3')
plt.ylabel('Brain Weight in grams')
plt.legend()
plt.show()
```

Fig. 4. Regression line creation

In the code, x and y variables are representing the coordinates of regression line. max_x is the

maximum value in the x-space and the min_x is the minimum value in in the x-space. So we get the change on the x axis. With using the linspace function program creates a vector with respect to change in x value. At the end program shows the regression line and data points in a graph.

```
# Calculating Root Mean Squares Error
rmse = 0
for i in range(n):
    y_pred = b0 + b1 * X[i]
    rmse += (Y[i] - y_pred) ** 2
rmse = np.sqrt(rmse/n)
print("RMSE")
print(rmse)
```

Fig. 5. Error rate calculation

For finding the estimated values B0+B1*X[i] executing for each data. After finding the estimated values, rmse is calculated with finding the square of each difference and summing them all. For last, rmse is divided to sample space size and square root has token.

```
# Calculating R2 Score
ss_tot = 0
ss_err = 0
for i in range(n):
    y_pred = b0 + b1 * X[i]
    ss_tot += (Y[i] - mean_y) ** 2
    ss_err += (Y[i] - y_pred) ** 2
r2 = 1 - (ss_err/ss_tot)
print("R2 Score")
print(r2)
```

Fig. 6. Coefficient of Determination.

At the last part, R^2 value is calculated. ss_tot is stands for total sum of squares and ss_err is total sum of errors. ss_tot uses the current average of y but ss_res uses predicted y value for each data point. By the way, program finds the predicted values and actual values. ss_tot shows us the amount of variation in the dependent variable. ss_res finds the differences between the actual Y and the predicted Y then squares it.

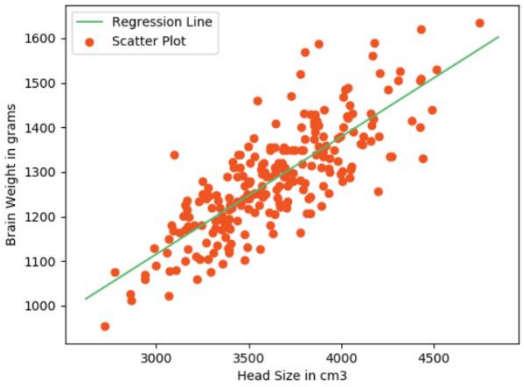

Fig. 7. Plot Output

First output is the plot of the best-fit line and data points. In the graph each data has a distance between the line. This distance is called as residual. While making the calculations for performance analysis, this distance will be used.

```
Coefficients
0.26342933948939945 325.57342104944223
RMSE
72.1206213783709
R2 Score
0.6393117199570003

Process finished with exit code 0
```

Fig. 8. Analysis Output

Second output represents the coefficient values and analysis rates. First one is stands for B0 and second one is B1. RMSE score is 72 that means not a bad model.

4.2. Sklearn Example

This example also uses the same dataset with first one. In the code train and test sets are declared. After spliting dataset into train and test part the regression function is executed. The main advantage is all funcions are declared in sklearn's model_selection library. The main functions used are fit and predict.

```
#fitting simple linear regression to the training
regressor = LinearRegression()
regressor.fit(x_train,y_train)

#predict the test result
y_pred = regressor.predict(x_test)

#to see the relationship between the training data
plt.scatter(x_train,y_train,c='red')
plt.show()

#to see the relationship between the predicted
#brain weight values using scattered graph
plt.plot(x_test,y_pred)
plt.scatter(x_test,y_test,c='red')
plt.xlabel('headsize')
plt.ylabel('brain weight')
plt.show()
```

Fig. 9. Linear Regression implementation

Code is an implementation of sklearn library for making linear regression. After calling LinearRegression() function, fit operation executed. With fit function x_train and y_train variables will be learned by the regressor. That means we've created our model and trained it. y_pred part is for making a prediction for the test dataset's x values. With the function, we can generate predicted values for a test cases with using train data. And the working regression model is created. The main point is regressor is trained with training data but when the predict function is added regressor didn't create a new equation. Regressor uses the train set's equation for making assumptions about test_y values. That code is more realistic for AI applications. At the end program prints the train dataset plot (Fig.10) and the test dataset that executed with regressor (Fig.11). After the execution of regression, another important point is to make the performance analysis. In the code, RMSE method is used. As a final result, RMSE is 72. That means the model is not bad at all. But not the best case. RMSE calculation is shown in Fig.12.

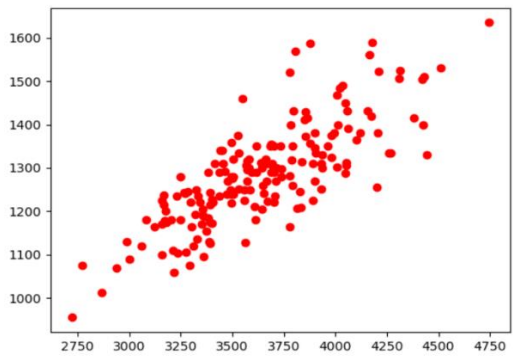

Fig.10. Train data plot

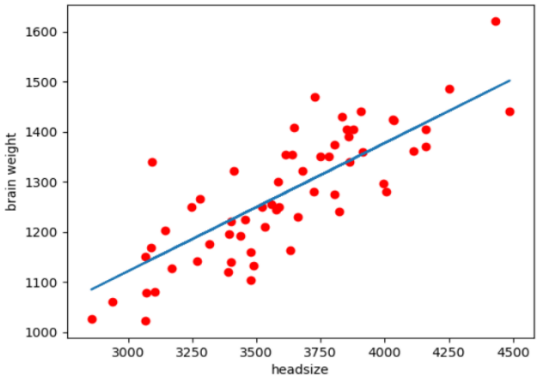

Fig.11. Best-fit line and test data

```
#combined rmse value
rss=((y_test-y_pred)**2).sum()
mse=np.mean((y_test-y_pred)**2)
print("RMSE=",np.sqrt(np.mean((y_test-y_pred)**2)))
```

Fig.12. RMSE calculation

5. Conclusion

Linear regression is one of the most used methods. The main advantage of linear regression is its comprehensible, simple and widespread usability. Linear regression is commonly used in statistical software and business intelligence tools.

6. References

[1] Jain, M. (2019). *Simple Linear Regression*. [online] Manishjaincloud.blogspot.com. Available at: http://manishjaincloud.blogspot.com/2019/05/simple-linear-regression.html

[2] Goswami, R. (2018, August). Root-Mean-Square Error (RMSE): Machine Learning. Retrieved from https://www.includehelp.com/ml-ai/root-mean-square error-rmse.aspx.

[3] Gandhi, R. (2018). *Introduction to Machine Learning Algorithms: Linear Regression*. [online] Medium. Available at: https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a