Linear Models for Regression and Classification

Seminar Data Mining

Leonardo Maglanoc
Department of Informatics
Technische Universität München
Email: maglanoc@in.tum.de

***Abstract*— Linear Models are a vast class of algorithms used for Data Mining. This paper looks at the two most important implementations of Linear Models: Multiple Linear Regression for predicting a continuous label and Binary Logistic Regression for datasets with two discrete labels. These are then applied to real repositories as case studies. In the first case study, the goal is to predict the overall happiness of a country with indicators such as GDP to understand Multiple Linear Regression better. In the second one, the goal is to predict if a breast mass sample is cancerous or not based on biological parameters with Binary Logistic Regression. In the end, Polynomial Regression and Neural Networks are presented as possible extensions for Linear Models to also account for nonlinear relationships.**

***Keywords*— Machine Learning, Data Mining, Supervised Learning, Linear Models, Multiple Linear Regression, Binary Logistic Regression**

I. Introduction

The goal of Data Mining is to extract knowledge of datasets to be able to understand the past or to predict the future when models are applied to unseen data. There exist many different algorithms to achieve this. A Data Scientist should have a good overview of all methods and know when they are most effective and when not to use specific algorithms.

Linear Models are essential tools in this regard, and this paper explains the details of two specific Linear Models: **Multiple Linear Regression** and **Binary Logistic Regression**. Understanding Linear Models is the first step in understanding all Data Analysis algorithms because they are the basis for more sophisticated models.

A. General Idea of Linear Models

Linear Models have the restriction that the input parameters have to be combined via a linear combination $a_1v_1 + a_2v_2 + \cdots + a_nv_n$. A simple linear combination is essentially Linear Regression, but there are also other Linear Models where the resulting linear combination is additionally transformed in a special way. In Logistic Regression, the linear combination is fitted to a particular nonlinear function.

B. General application in the scope of Data Mining

Table 1 shows an excerpt of a survey conducted by a popular Data Science Website called Kaggle[1]. According to the survey, Linear Models are not only the most frequently applied Data Analysis algorithm but they are used in almost all repositories. There are good reasons why simpler models are preferred to more complex models, despite being less generalizable. For example, Neural Networks are proven to be general function approximators as shown in [1]. Due to their simplicity, Linear Models are often the first algorithms to try out on a dataset. Even if the model's accuracy is not as good as it could theoretically be with a better model or the predictive ability is lacking, it can be used as an effective tool for Data Exploration to understand the relationships between variables better. This will be shown in the case studies.

Algorithm	in how many Kaggle repositories
Linear or Logistic Regression	83.7%
Convolutional Neural Networks	43.2%
Recurrent Neural Networks	30.2%
Generative Adverserial Networks	7.2%

Table I
excerpt from Kaggle Survey 2020

In summary, before more advanced algorithms such as Neural Networks are applied to solve a Data Mining related problem, it is better to try simpler models like Linear or Logistic Regression.

C. Difference between Regression and Classification

Before the specific algorithms can be explained, one needs first to understand the different problem settings each algorithm tries to solve. Regression and Classification are problems that can be categorized under the field of Supervised Learning. In Supervised Learning, the goal is to predict unknown data points with the help of an already existing labeled dataset. Classification means that the predicted label of the model is a discrete set, often called groups. The most straightforward case of a Classification problem is when there are only two labels. Most of the time, the label is "true" (1) or "false" (0). One problem could be to classify the species of a bird given some information like wing length and weight. In contrast to that, models for Regression predict a continuous label. One example could be, given the price of a house in Munich, predict the size of the house in square meters. Figure 1 illustrates the difference between Regression (right) and Classification (left) with Linear Models. The left plot has two continuous input parameters, and the two groups are encoded as colors. A Classification Model would try to find a line such that the two colors are separated from each other. The right plot has one

[1] https://www.kaggle.com/kaggle-survey-2020

continuous input and output parameter. A Regression Model tries to follow the trend of the points as well as possible.

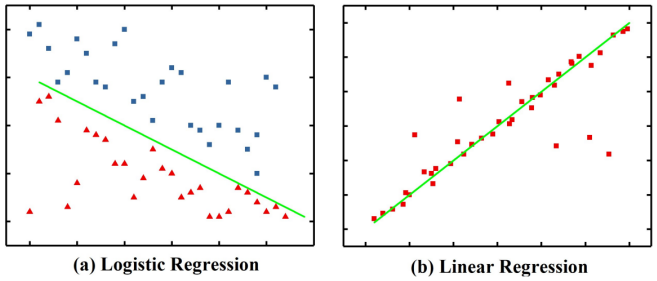

Fig. 1. difference between Classification and Regression (adapted from [2])

II. Multiple Linear Regression

This section introduces the first algorithm, **Multiple Linear Regression**. With this model, one can predict continuous quantities. For a more thorough description, Hastie's et al. *The Elements of Statistical Learning* [3, pp. 43–94], Bishop's *Pattern Recognition and Machine Learning* [4, pp. 137–173] or Yan's et al. *Linear Regression Analysis* [5] can be used as a reference.

A. Mathematical description

For one predicted data-point y_i, corresponding explanatory data-parameters $\boldsymbol{x}$, parameters of the model $\boldsymbol{\beta}$ and an error ϵ_i, the model can be written as:

$$y_i = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip} + \epsilon_i = \boldsymbol{x}_i^T \boldsymbol{\beta} + \epsilon_i$$

With $\boldsymbol{x}^T$ as the transposed x-vector and $\boldsymbol{\beta}$ as β-vector, it can be formulated in matrix form in the following way:

$$\boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix},$$

$$X = \begin{pmatrix} \boldsymbol{x}_1^T \\ \boldsymbol{x}_2^T \\ \vdots \\ \boldsymbol{x}_n^T \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix}$$

$$\boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix}, \quad \boldsymbol{\epsilon} = \begin{pmatrix} \epsilon_0 \\ \epsilon_1 \\ \vdots \\ \epsilon_n \end{pmatrix},$$

in a compact notation:

$$\boldsymbol{y} = X\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{1}$$

The goal is to minimize the error:

$$\boldsymbol{\epsilon} = \boldsymbol{y} - X\boldsymbol{\beta} \tag{2}$$

Figure 2 visualizes a Regression Plane with two input parameters. The geometric interpretation of the error in equation 2 is that the error is the distance between observed and fitted data points (see figure 2, the white and black data points).

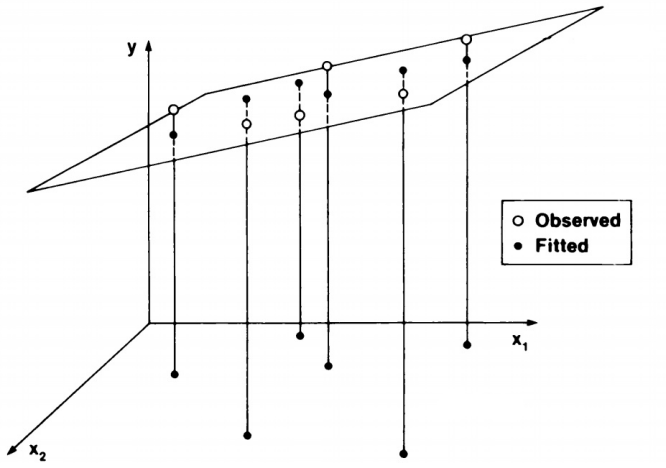

Fig. 2. geometric representation of a Linear Regression Plane (adapted from [6])

B. Ordinary Least Squares

The most popular method to minimize the error is via **Least Squares**, which minimizes the squared residuals of each predicted value from the dataset and the values the model produces. This can be reframed as a minimization problem:

$$\arg\min_{\boldsymbol{\beta}} \left[(\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta}) \right] \tag{3}$$

By utilizing the partial derivatives, one arrives at the following linear system of equations:

$$X^T X \boldsymbol{\beta} = X^T \boldsymbol{y} \tag{4}$$

As seen in the formula above, any algorithm which can solve a Linear System of Equations, can be applied. Alternatively, the direct solution can be calculated with this formula via the inverse of a matrix:

$$\boldsymbol{\beta} = \left(X^T X\right)^{-1} X^T \boldsymbol{y} \tag{5}$$

C. Assumptions

Linear Regression has two main assumptions. These need to be fulfilled for the model to be as effective as possible with predictions of previously unknown data.

Firstly, it is assumed that there are no multicollinearities between the input parameters. The input parameters need to be linearly independent because of the linear constraint we are setting on the model. When this assumption is false, the easiest case is if one parameter is precisely equal to another

parameter. There are more sophisticated Linear Regression Algorithms, where the model can be fitted even if there are hidden collinearities. One such algorithm solves this problem by iterating over subsets of input parameters and determining the best subset with as little multicollinearities as possible.

Secondly, it is assumed that the errors are Gaussian distributed with zero mean and constant variance.

If both assumptions are applied to the so called Maximum-Likelihood-Method, taking the minimum of the squared residual sums is equal to Linear Regression being the most probable model for explaining the observed data, which justifies using Multiple Linear Regression with Least Squares, if the assumptions are accurate. The proof for the Maximum-Likelihood of Linear Regression can be found in [5] in chapter 2.4.

D. Evaluation of the model

To be able to evaluate a Regression Model, one needs to look at **residuals**, which is the distance between the data points predicted by the model and the actual data points (see equation 2). One metric, which can always be used, is the **Mean Squared Error** (MSE). It can be interpreted geometrically as summing up the squared distance of all residuals. However, R^2 is the most often used metric for Linear Regression. It ranges from zero to one and describes how well the model predicts the data points. $R^2 = 1$ means the model perfectly fits the dataset. It can be calculated in the following way:

First, you need the **mean** of the data-points,

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{6}$$

the **total sum of squares**, which is proportional to the variance of $\boldsymbol{y}$,

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \tag{7}$$

the **residual sum of squares**, which was introduced in equation 2,

$$SS_{res} = (\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta}) = \sum_i \epsilon_i^2 \tag{8}$$

R^2 can then be calculated via

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \tag{9}$$

However, there are some problems, which cannot be resolved by only looking at this metric. R^2 never decreases by adding new input parameters to the Linear Regression Model. It might even be that adding non-sensical parameters results in a higher R^2-Value, due to random noise in the data. There are some ways to mitigate this. One solution is to change the minimization problem, such that high-valued parameters are more punished. As another option, the R^2 metric can be adjusted.

There are also alternative metrics: The F-test evaluates if the input parameters are significant via the following null and alternative hypothesis:

$$H_0 : \boldsymbol{\beta} = \mathbf{0} \quad H_1 : \beta_i \neq 0 \text{ for some i} \tag{10}$$

E. Computation

The direct computation of the model parameters is not common because computing the inverse of a matrix is not numerically stable. One method to avoid computing the inverse is to apply an iterative algorithm, which converges to a minimum like Gradient Descent. Due to the convexity of the error function, the iterations are guaranteed to converge to the global minimum. However, the more common way is via the Singular Value Decomposition (SVD) [6], which is also used by Scikit-Learn [7], a popular library for Data Analysis. It has a complexity of $O(n_{\text{samples}} n_{\text{features}}^2)$ as long as $n_{\text{samples}} \geq n_{\text{features}}$. The parameters can be calculated in the following way:

$$\begin{aligned} \boldsymbol{y} &= X\boldsymbol{\beta} + \boldsymbol{\epsilon} = U\Sigma V^T\boldsymbol{\beta} + \boldsymbol{\epsilon} \\ &\Rightarrow \boldsymbol{\beta} = V\Sigma^{-1}U^T y \end{aligned} \tag{11}$$

U $(n \times n)$ and V $(p \times p)$ are orthogonal matrices and Σ $(p \times p)$ consists only of diagonal entries known as the singular values of X.

SVD can be seen as a generalization of the eigenvalue decomposition, which works for all complex matrices, not only for diagonalizable ones. With the matrix-decomposition, the problem reduces to simple matrix-vector multiplication. In addition, the properties of the model can be seen quickly by looking at the Σ-Matrix. Singular values, which are zero or near-zero, indicate high collinearities between the particular parameters.

F. Case Study

The World Happiness Report[2] is the accumulation of different surveys, which were conducted for finding out how people would rate their own life in regards to happiness. The people are then grouped into nationally representable samples for one particular country to be able to rate a happiness score for a country. In the provided dataset from 2021, country metrics (see figure 3) are included to understand better how they might influence a country's overall happiness.

	Country name	Ladder score	Logged GDP per capita	Social support
0	Finland	7.842	10.775	0.954
1	Denmark	7.620	10.933	0.954
2	Switzerland	7.571	11.117	0.942
3	Iceland	7.554	10.878	0.983
4	Netherlands	7.464	10.932	0.942

Fig. 3. excerpt from Happiness data-set

The Regression Model will try to predict the happiness score based on logged GDP per capita x_1, social support x_2, healthy life expectancy x_3, freedom to make life choices x_4, generosity x_5 and perceptions of corruption x_6.

[2] https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021

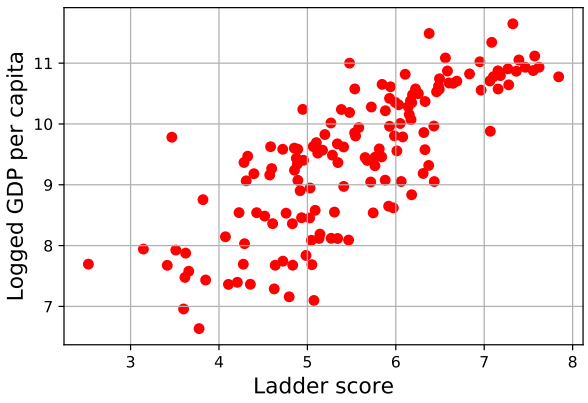

Fig. 4. scatterplot of ladder score and gdp

Before training the model, it is good to assess if the assumptions for Linear Regression are fulfilled. The input- and output variables should have a linear relationship with approximately Gaussian distributed and independent errors. Via a scatter plot between all pairs of input- and output parameters, the assumptions can be approximately evaluated visually. One possible scatter plot between ladder-score and GDP is displayed in figure 4. The model was trained with Statsmodels [8], a popular library for statistical methods.

OLS Regression Results

Dep. Variable:	Ladder score	R-squared:	0.756
Model:	OLS	Adj. R-squared:	0.746
Method:	Least Squares	F-statistic:	73.27
Date:	Sun, 20 Jun 2021	Prob (F-statistic):	5.06e-41
Time:	16:27:58	Log-Likelihood:	-116.50
No. Observations:	149	AIC:	247.0
Df Residuals:	142	BIC:	268.0
Df Model:	6		
Covariance Type:	nonrobust		

Fig. 5. description of Regression Model

The resulting model is:

$$y = -2.2372 + 0.2795x_1 + 2.4762x_2 + 0.0303x_3 + 2.0105x_4 + 0.3644x_5 - 0.6051x_6 \quad (12)$$

The R^2-value of the model with 0.756 in figure 5 indicates good performance. Now it would be good to go back and try to explain why $R^2 \neq 1$. This means the Linear Regression Model does not fully explain the dataset. Possible reasons for this are measurement errors or nonlinear relationships. If you incorporate knowledge about the domain subject, a saturation effect with GDP in regards to happiness, as might be seen in figure 4, could be one possible explanation for a nonlinear relationship: When you reach a certain GDP-threshold, more GDP results in not as much extra happiness as with lower levels. An alternative interpretation is a hard limit to how much happiness one can reach with money alone.

III. Classification with Binary Logistic Regression

In **Binary Logistic Regression**, only two classes are categorized, a "true" (1) class and the corresponding "false" (0) class. For a more thorough description, Jurafsky's et al. *Speech and Language Processing* [9, pp. 76–95], Bishop's *Pattern Recognition and Machine Learning* [4, pp. 179–220] or Hastie's et al. *The Elements of Statistical Learning* [3, pp. 101–135] can be used as a reference.

A. Mathematical description

The key idea of Logistic Regression is to reuse the previously presented Linear Regression method to predict the probability for belonging into the "true" class. This is also the reason why it is called Logistic *Regression*. The problem with this idea is that Linear Regression has a range of $(-\infty, +\infty)$ while probabilities have a range of $[0, 1]$. To solve this problem, a transformation function is needed, which maps from $(0, 1)$ to the real numbers like the Logit Function $y = \log\left(\frac{x}{1-x}\right)$.

A linear relationship between the log-odds and the input parameters is assumed. Reusing the model from equation 1, it can be written as follows:

$$\log\left(\frac{p}{1-p}\right) = X\boldsymbol{\beta} \quad (13)$$

By algebraic manipulation, it can be reduced into the following function:

$$P[y = 1|\boldsymbol{x}] = \frac{1}{1 + e^{-(\boldsymbol{\beta}^T \boldsymbol{x})}} = \sigma(\boldsymbol{\beta}^T \boldsymbol{x}) \quad (14)$$

The Model P can be interpreted as the probability of the object being in the "true" group conditioned on the input parameters. Figure 6 illustrates the difference between Linear and Logistic Regression with the black points as observed data points. Linear Regression (blue plot) fits a straight line or plane, whereas Logistic Regression (red plot) fits a nonlinear function. The reason for this is that a new nonlinear term is added in addition to a linear combination (see equation 14). The effect of the nonlinear part is to restrict the range of the function to $(0, 1)$.

The current Regression problem can be turned into a Classification problem, by setting a decision boundary value:

$$y = \begin{cases} 1 & \text{if } P(y = 1|\boldsymbol{x}) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \quad (15)$$

Depending on the context of the problem, the decision boundary can be changed to a higher or lower value. For example, a cancer recognition software would set a lower threshold because a false negative would be more unfortunate than a false positive. A false negative in this specific case would mean that the person actually has cancer. However, the model did not accurately identify the sickness, which means it would go undetected for a longer time.

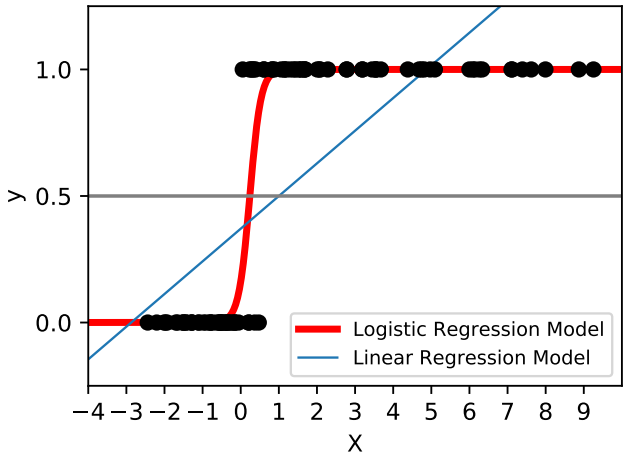

Fig. 6. difference between Linear and Logistic Regression

The error between predicted and actual value can be calculated in the following way via the cross-entropy loss:

$$e(\hat{y}, y) = -\Big[y \log \sigma(\boldsymbol{\beta}^T \boldsymbol{x}) + (1-y)\log(1-\sigma(\boldsymbol{\beta}^T \boldsymbol{x}))\Big] \quad (16)$$

The goal is to find the set of weights $\boldsymbol{\beta}$ such that the loss function averaged over all examples is minimized with $f(\boldsymbol{x}^{(i)}; \boldsymbol{\beta})$ being the predicted value of Logistic Regression evaluated at $\boldsymbol{x}^{(i)}$:

$$\arg\min_{\boldsymbol{\beta}} \frac{1}{m} \sum_{i=1}^{m} e(f(\boldsymbol{x}^{(i)}; \boldsymbol{\beta}), y^{(i)}) \quad (17)$$

There exists no general closed formula. Instead, Gradient Descent can be used to find a better solution iteratively. Due to the convexity of the error function, Gradient Descent is guaranteed to find the global minimum of the error function.

B. Evaluation of the model

The standard way of evaluating a Classification model is via splitting the dataset into two different sets, one for fitting the model, which is called the **training dataset**, and one for testing the model, also called the **testing dataset**. Then it is evaluated how many testing-data-points it can correctly classify, which is called the **Classification Accuracy** and is given in percentage.

Alternative metrics similar to those from Linear Regression can also be reused for Logistic Regression. However, they are not as effective because they were designed with traditional Regression in mind.

C. Case Study

Figure 7 shows an excerpt of the cancer-dataset[3] from 1995. Cancerous and non-cancerous breast mass was sampled. The features of the cancer-dataset describe characteristics of the cell nuclei. Each data point is categorized into benign (B) and malignant (M), denoted in the diagnosis column. The goal is to predict if the sample is malignant or not based on some numeric input parameters.

[3] https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

	diagnosis	texture_mean	perimeter_mean	smoothness_mean
564	M	22.39	142.00	0.11100
565	M	28.25	131.20	0.09780
566	M	28.08	108.30	0.08455
567	M	29.33	140.10	0.11780
568	B	24.54	47.92	0.05263

Fig. 7. excerpt from cancer-dataset

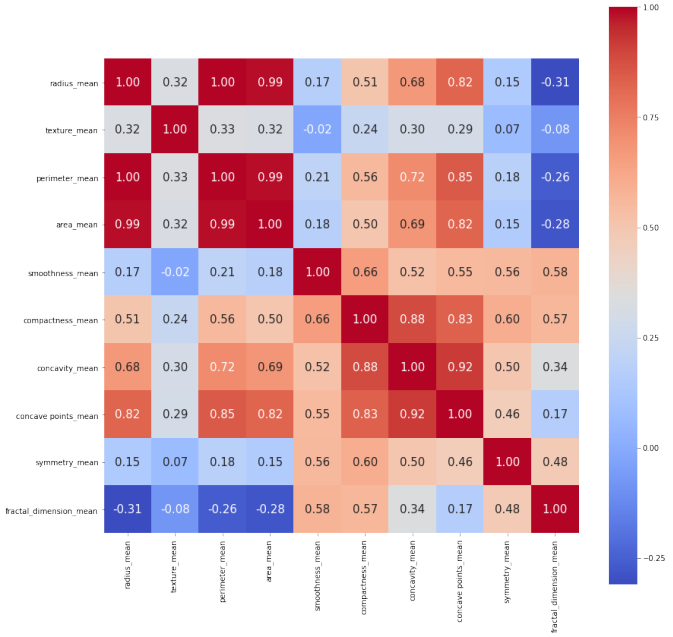

Fig. 8. correlation matrix of features

Looking at the correlation matrix between the parameters in figure 8, those variables with as close to zero pairwise correlation as possible should be chosen for an effective model due to the assumption of the input parameters being linearly independent of each other.

For the model, texture mean x_1, perimeter mean x_2, smoothness mean x_3, compactness mean x_4 and symmetry mean x_5 were chosen as input parameters to predict if the cancer was malignant or not.

The dataset was split into a training set to train the model, which amounts to 70%, and a validation set to test the model's performance, which accounts for 30% of the data. Looking at the metrics of table 2, the model reached a good performance with an accuracy of 91.2%. This means that the model only misclassified a tenth of the validation data points.

The resulting model is:

Metrics of the model	Values
Classification Accuracy	91.2%
Mean Squared Error	0.0877

Table II
evaluation of the Logistic Regression Model

$$P[y=1|\boldsymbol{x}] = \frac{1}{1+e^{-(0.303x_1+0.199x_2+73.3x_3+12.1x_4+27.3x_5)}} \quad (18)$$

IV. Summary

This paper is about how Linear Models are used for Data Mining purposes. The main idea of Linear Models is that the input parameters are combined via a linear combination. Firstly, Multiple Linear Regression was introduced. With this method, one constructs a continuous function. An exact closed formula is used to determine the model with the least error given the dataset. Some assumptions are made for the model to work well. There should be no collinearities, and the errors should be independent and nearly Gaussian distributed. Linear Regression was applied as a case study to predict the happiness of different countries based on GDP and other indicators.

The second part of the paper was about Binary Logistic Regression, which can be seen as a modification of Linear Regression. Binary Logistic Regression aims to predict the probability of one object being in the "true" or "false" group. To reuse Linear Regression for predicting the probabilities, the Logit-Transformation is applied because it maps from $(0,1)$ (the range of possible probabilities) to $(-\infty,\infty)$ (the range of Linear Regression). This can then be turned into a Classification solution by setting everything above a probability threshold to one group and vice-versa. A closed formula does not exist, but due to the convexity of the error function, the iterative Gradient Descent Algorithm can be used, as it is guaranteed to converge to the global minimum. Logistic Regression was applied to predict if a breast mass sample is cancerous or not based on numeric features of the cell nuclei.

Both algorithms are important tools of a Data Scientist, as they can be used to solve various problems. Before more sophisticated models are applied, it is good to test simpler models such as these. This is why Linear Models are so popular and why they are used in almost all repositories on the Data Science website Kaggle.

V. Outlook

As seen in the case studies, Linear Models can be very effective given certain circumstances. However, it is easy to construct counter-examples where they fail. Their main drawback is not accounting for nonlinear interactions.

A. Polynomial Regression as extension for Linear Regression

The easiest way of adding nonlinear predictions to Linear Regression is via Polynomial Regression. For this to work, the matrix X needs to be revised in the following way:

$$X' = \begin{pmatrix} 1 & x_1 & \cdots & x_1^p \\ 1 & x_2 & \cdots & x_2^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \cdots & x_n^p \end{pmatrix} \quad (19)$$

Previously, p denoted how many input parameters the Linear Regression Model has. To reuse the old formulas (equations 1 to 5 and 11), we need to restrict ourselves to only one input parameter. Instead of different input parameters, monomials of the form $1, x, \cdots, x^p$ are added for one input parameter x. With this new matrix X' from equation 19, the same formulas and methods from Multiple Linear Regression can be reused. Furthermore, other basis functions can be applied. Any set of basis functions, not only polynomials, can be used, as long as the functions are combined via a linear combination as shown in [4, pp. 138–140].

B. Logistic Regression as a building block for Neural Networks

Binary Logistic Regression can be interpreted as a One-Layer Perceptron according to [10], which is a more straightforward implementation of a Neural Network with only one Neuron and with the Logistic Function as the Activation Function as illustrated in figure 9.

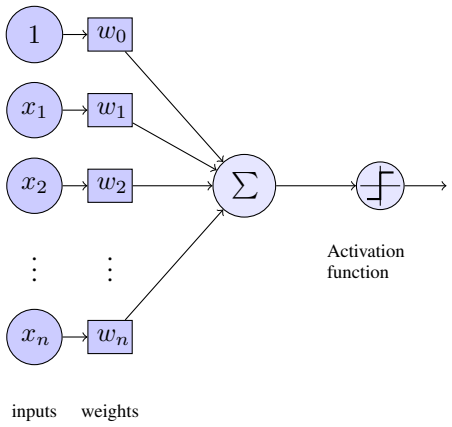

Fig. 9. Logistic Regression as a Neuron of a Neural Network

Many aspects mentioned about Logistic Regression in this paper can be directly applied to more general Neural Networks. With the extensive form of the Neural Network Model, favorable properties of Logistic Regression are lost. In particular, the minimization problem is more complex because the error function is not convex anymore. Traditional Gradient Descent is not guaranteed to converge to the global minimum anymore. However, one significant advantage of Neural Networks in comparison to Logistic Regression is their mathematically proven ability to approximate any continuous function as shown in [1].

REFERENCES

[1] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0893608089900208

[2] G. Y. Y. Q. Xianjin Fang, Fangchao Yu, "Regression analysis with differential privacy preserving," *IEEE Access*, vol. 7, pp. 129 353–129 361, 2019.

[3] J. F. Trevor Hastie, Robert Tibshirani, *The Elements of Statistical Learning*. Springer, 2008.

[4] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[5] X. G. S. Xin Yan, *Linear Regression Analysis*. World Scientific, 2009.

[6] J. Mandel, "Use of the singular value decomposition in regression analysis," *The American Statistician*, vol. 36 No.1, pp. 15–24, 1982.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] S. Seabold and J. Perktold, "statsmodels: Econometric and statistical modeling with python," in *9th Python in Science Conference*, 2010.

[9] J. H. M. Daniel Jurafsky, *Speech and Language Processing*. Third Edition Draft, 2020.

[10] "Neural net blogpost," https://sebastianraschka.com/faq/docs/logisticregr-neuralnet.html, accessed: 2021-22-05.