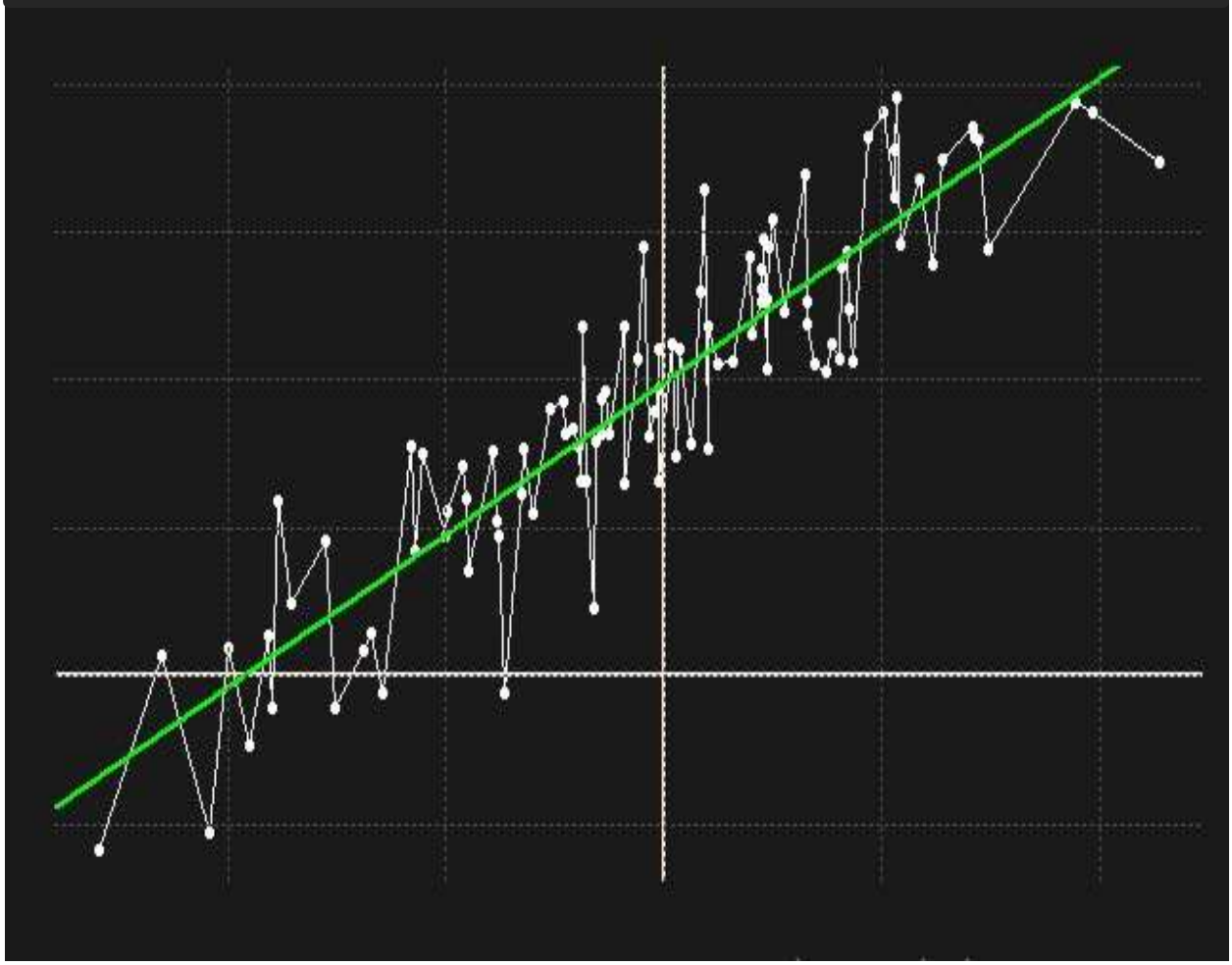
“NOISE-ADDED” REGRESSION MODELS.

- **Stanley Sayianka**



Noise-added regression models – a research and project

Written by Stanley Sayianka as a research product for submission to the product development webinar of The Actuarial Students' Society of Kenya.

The theme of the product development and design competition is **Actuarializing the world with Sustainable Development Goals**, and one way in which this could be accomplished is through the following:

- Decent Work and Economic Growth: Creating jobs for all to improve living standards, providing sustainable economic growth
- Industry, Innovation and Infrastructure: Generating employment and income through innovation.
- Reduced Inequalities: Reducing income and other inequalities, within and between countries

Since actuaries and actuarial work is regarded as one of the most data driven roles, actuaries rely on data, analytics and information acquired from statistical models to inform their decisions which are in turn tied directly to the performance and well-being of businesses, industries, lives of people and finances.

Thus improving on the work of these statistical models actually leads to improved decision making by the actuaries, and this may in turn lead to bettering of the lives of many people, businesses, industries and governments.

Abstract

Regression models are one of the most widely used models in actuarial work in the context of predictive analytics, since despite their simplicity, their performance competes with some of the most complex models available in predictive modelling literature. At its simplest form, the core of regression models is to seek for a linear relationship between a dependent variable (often called the response variable) and an independent variable, also known as the predictor variable. Although the relationship between the variables need not necessarily be linear, the parameters of linear regression models are linear, thus the name. Regression models can have diverse interpretations and inference, but as of now I would focus on the following: a regression model can be viewed in terms of the total variation in the response variable that is explained by the model, and that which is unexplained by the model. It would be desirable that the amount of variability explained by the model is much higher, and that which is unexplained is much lower. The aim of this research and project paper is to improve the linear regression models predictive performance, based on incorporating the unexplained variation in its predictive component, using the nearest neighbor algorithm, thus the term "noise-added". The experimental result shows that the noise added-regression models perform far better than the normal regression models, using performance metrics such as the Mean square error (MSE) and the root mean square error (RMSE). The results differ based on the number of nearest neighbor (k) used, although for significantly large k ($k > 10$), results are similar.

Keywords: K-nearest neighbor, KNN, MSE, RMSE, noise, distance metric, normal distribution, error terms.

1. Introduction

Regression models find a lot of application in actuarial work, ranging from: forecasting and financial analysis, finding correlation between stock prices and currencies, valuing financial performance of a company with the performance of a particular index or industry, in capital asset pricing models, they are used to calculate the beta of a company, in sales e.g. of insurance products and policies regression can be used.

The most common similarity in linear regression models is the assumption made, which include: that the error terms are normally distributed, with a mean 0 and variance σ^2, which implies that the response variable is also normally distributed. The regression parameters commonly denoted β_i, have unbiased estimates, which are estimated using Bayesian or Maximum Likelihood estimation, such as the Least squares method (most common), or the weighted least squares. If the data at hands exhibits the above distribution properties, the linear regression models have reasonably impressive performance, but if the parametric assumptions are violated or not met, then the regression models have poor performance. Under inference in regression models, we seek to find:

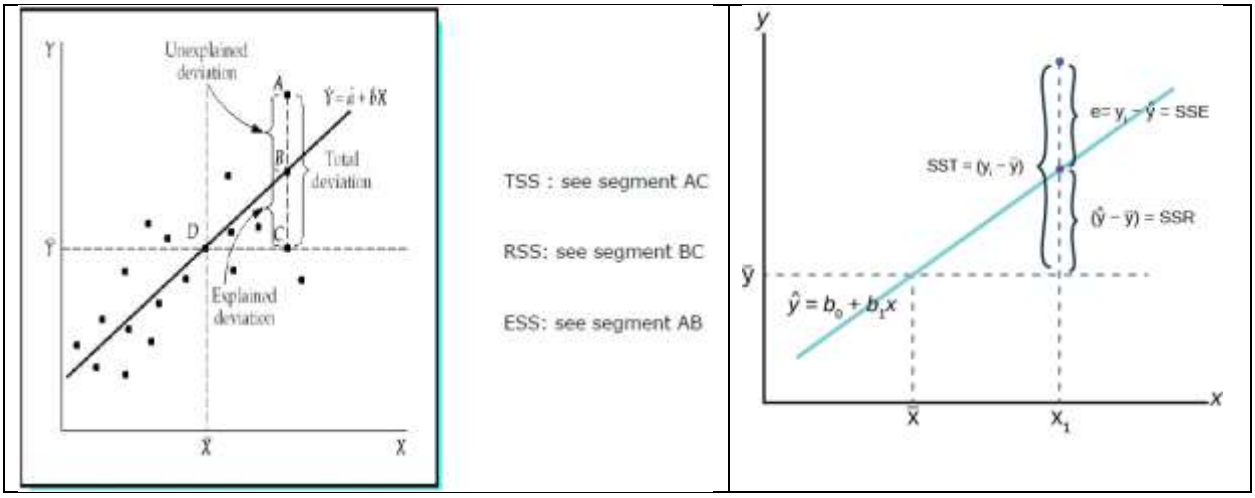
SS_{total} – Total Sum of Squares

$SS_{regression}$ – Regression Sum of Squares

$SS_{residual}$ – Residual Sum of Squares.

The total sum of squares equals the regression plus residual sum of squares, and it is normally desirable that models exhibit a regression sum of squares that is significantly higher, while a residual sum of squares that is significantly lower, since it represents the amount of variation unexplained or unaccounted for by the model.

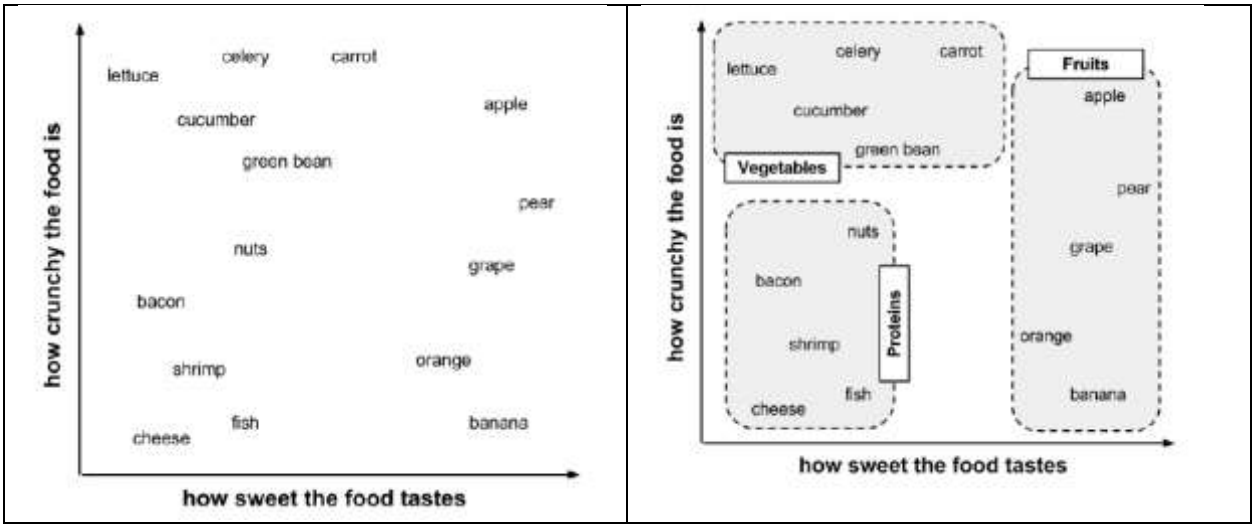
The regression and residual sum of squares can be illustrated using simple linear regression as follows:



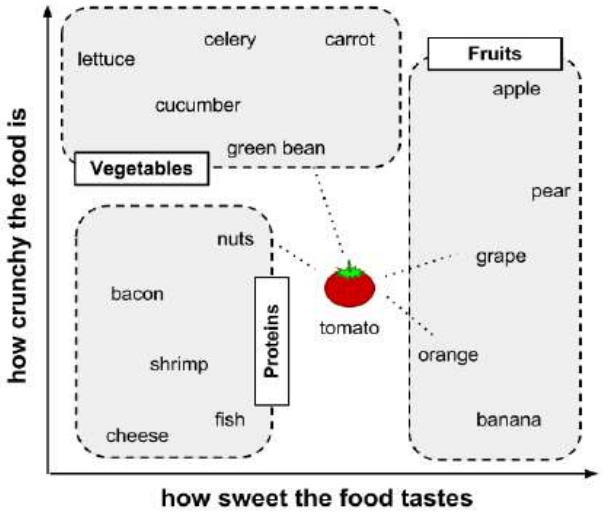
The nearest neighbor algorithm is an important algorithm specifically in classification and clustering literature in machine learning, which seeks to associate a particular data point as being of a particular class or cluster based on its distance to its closest data point. That is, the algorithm starts by computing distances between a single point and all the other points in a training set, the based on a specific chosen parameter k (the number of neighbors), a vote is taken, and the data point is the associated with the class or cluster that won the vote. The type of distance measure is of importance, but on most common distance measure that thrives in a lot of situations is the Euclidean distance measure (relies on Pythagoras theorem).

This can best be illustrated using an example:

In trying to categorize whether a tomato is a fruit or a vegetable, we can collect data on several fruits and vegetables as well, say on characteristics such as sweetness and crunchiness (on a scale of 1 to 10) as shown:



Since we notice that similar foods tend to be grouped together, such as vegetables can be seen on the top right and fruits on the uttermost left, then classifying the tomato as a fruit or vegetable can easily be done by calculating its distance (of crunchiness and sweetness) to the rest of the data above, and then the tomato can be classified as a fruit or vegetable depending on the class of its nearest neighbor(s).



ingredient	sweetness	crunchiness	food type	distance to the tomato
grape	8	5	fruit	*sqrt((6 - 8)^2 + (4 - 5)^2) = 2.2*
green bean	3	7	vegetable	*sqrt((6 - 3)^2 + (4 - 7)^2) = 4.2*
nuts	3	6	protein	*sqrt((6 - 3)^2 + (4 - 6)^2) = 3.6*
orange	7	3	fruit	*sqrt((6 - 7)^2 + (4 - 3)^2) = 1.4*

Thus based on this analysis, the nearest food to the tomato (in terms of crunchiness and sweetness) is the orange, and thus since an orange is a fruit, then the tomato qualifies to be a fruit, based on its first nearest neighbor. In choosing the three nearest neighbors, the orange, grape and nuts are the first, and since two out of the three are fruits, then the tomato qualifies to be a fruit. Settled!

In this review we seek to improve the performance of the predictive regression models by using the nearest neighbor approach, such that, we have training data which is used to build the regression model, then for each new observation in a test dataset (previously unseen data), the regression model is used to predict the response variables, and then the nearest neighbor algorithm calculates the nearest neighbors of each particular data observation of the test dataset from among all the data points in the training set (say k neighbors), then once the nearest neighbors have been identified, the residuals (noise terms) of those nearest neighbors are averaged, and that result is added to the predicted response of the test set data point.

The intuition behind this "noise-added" model is that, since in many cases there always exists a substantial amount of residual sum of squares (that is: amount of variation unaccounted for by the model), then the task of the regression model in a predictive sense is to explain the response variable in terms of the predictor variables, and the task of the nearest neighbor algorithm is to compensate for the unexplained variation in the response variable by the regression model, so in a way, all the variability is explained, that is:

Regression model – Accounts for the Regression sum of squares

Nearest neighbor algorithm – Accounts for the residual sum of squares.

2. Nearest neighbor search and distance measures

Since the nearest neighbor algorithm works by calculating distances, this implies that we should take into account issues such as:

i. Distance metrics – There are a lot of various distance metrics in literature, which all have teir strengths and weaknesses and contribute differently, thus the choice of a distance metric is critical for top performance. Common distance metrics include: Euclidean distance, Manhattan distance, LPnorm distance and Chebyshev distance, among others.

ii. The data types of our variables – This is also important since all these distance metrics work on numerical data only, although there exists string similarity distances (which are not of importance to us at the moment. The types of datasets in an actuarial set up mostly comprise of categorical variables, such as in estimating the medical bill to charge (claim), we could take into account several factors as (numerical: age, BMI) as well as (categorical: smoker status, region, gender), it would be of great effect if we decide to omit such important categorical calculations in our nearest neighbor search, thus we must find a way of integrating the categorical variables in our nearest neighbor search.

iii. Scaling – Scaling might be an issue in dealing with the nearest neighbor algorithm, since most distance metrics work well in standardized or normalized data, and thus there might be a need to scale the data appropriately before applying the nearest neighbor search.

Distance measures

As for the distance metrics to be considered, the Euclidean distance metric has been proven to be of great importance in majority of datasets, thus it has gained approval as one of the best distance metrics to be considered in nearest neighbor search, also due to the fact that it is one of the most intuitive distance (a

physical distance). It is generally preferred as the best distance metric to use in numerical continuous datasets.

Abbrev.	Name	Definition
MD	Manhattan	$\sum_{i=1}^{n} \lvert x_i - y_i \rvert$
CD	Chebyshev	$\max_i \lvert x_i - y_i \rvert$
ED	Euclidean	$\sqrt{\sum_{i=1}^{n} \lvert x_i - y_i \rvert^2}$

Dealing with categorical variables

As for the categorical variables in datasets, there would be a huge amount of information loss if we decide to altogether abandon the categorical variables of a dataset during the nearest neighbor search, thus two options come to mind on integrating them:

i. We could decide to convert the categorical datasets into numerical or binary codes, a process commonly referred to as dummy coding or one hot encoding. This can best be illustrated as shown below:

If our dataset contains a categorical variable say Gender, which includes the unique values: Male and Female, then we could convert these into dummy codes as shown:	Create a variable say Gendercode, which is numerical (binary) as shown: $$Gender\ code = \begin{cases} 1 & if\ Male \\ 0 & if\ female \end{cases}$$
If our dataset contains a categorical variable say Region, which includes the unique values: Nairobi, Mombasa and Kisumu, then we could convert these into dummy codes as shown:	Create two Regioncode variable which is numeric (binary) as shown: $$RegionNairobi = \begin{cases} 1 & if\ Nairobi \\ 0 & if\ others \end{cases}$$ $$RegionMombasa = \begin{cases} 1 & if\ Mombasa \\ 0 & if\ others \end{cases}$$
Most commonly, when we have a categorical variable with many variables, we could employ one hot encoding, say if we have a categorical variable on colors such as: "red" "red" "green" "red" "blue" "red" "green" "red" "green" "blue" "green"	Then one hot encoding would create variables equal to the unique colors as shown:

	red	green	blue
1	1	0	0
2	1	0	0
3	0	1	0
4	1	0	0
5	0	0	1
6	1	0	0
7	0	1	0
8	1	0	0
9	0	1	0
10	0	0	1
11	0	1	0

An advantage of dummy coding and one hot swapping is that it transforms the categorical variables into numerical (binary) variables, whereby the distances can then be calculated, using appropriate distance metrics for binary variables such as the Manhattan distance and the Chebyshev distance metric, and the result obtained to be added on the calculated Euclidean distance of the numerical variables.

ii. Another option of dealing with the categorical variables is using splitting functions. In this case, the dataset is split into sub datasets based on the unique values of a particular categorical variable, in order to create sub datasets equal to the unique number of classes of a categorical dataset. For example, in the case of a Gender (Male, Female) variable in a dataset, the dataset will be split into two sub datasets, the male dataset and the female dataset, thus eliminating the need to include the gender variable in the datasets. Therefore, distances will be calculated based on the gender only, such that for a new test instance, we first check the gender: if the gender is Male, we only do a nearest neighbor search in the male dataset, as shown below:

Original dataset

	X1	X2	X3	X4	X5	Gender
1	0.4743832	-2.135426963	0.06357342	0.2957151	1.705609595	Female
2	-0.8736627	0.407277239	-0.12378747	-0.9533699	0.693084477	Female
3	-0.5831292	0.287344236	0.61170611	-0.2960975	-1.084750491	Female
4	0.3243549	-0.109357021	0.54145782	0.1691675	-0.698589128	Female
5	-0.9543695	-0.102396060	-0.19074689	1.6933941	-1.078875322	Male
6	1.4648694	-0.928958536	1.22207088	1.3330054	-0.667178498	Female
7	0.5623642	-0.196721527	-1.06059130	1.5593659	0.003488238	Male
8	-1.0540993	-0.617370886	-3.31318406	-0.3421785	-0.849211781	Male
9	1.8594768	0.001853609	0.34846078	-1.5367804	-0.986384194	Female
10	1.3501537	-0.080182767	-0.44378898	-1.2307136	0.430773629	Male

Split Male dataset

	X1	X2	X3	X4	X5
5	-0.9543695	-0.10239606	-0.1907469	1.6933941	-1.078875322
7	0.5623642	-0.19672153	-1.0605913	1.5593659	0.003488238
8	-1.0540993	-0.61737089	-3.3131841	-0.3421785	-0.849211781
10	1.3501537	-0.08018277	-0.4437890	-1.2307136	0.430773629

Split Female Dataset

	X1	X2	X3	X4	X5
1	0.4743832	-2.135426963	0.06357342	0.2957151	1.7056096
2	-0.8736627	0.407277239	-0.12378747	-0.9533699	0.6930845
3	-0.5831292	0.287344236	0.61170611	-0.2960975	-1.0847505
4	0.3243549	-0.109357021	0.54145782	0.1691675	-0.6985891
6	1.4648694	-0.928958536	1.22207088	1.3330054	-0.6671785
9	1.8594768	0.001853609	0.34846078	-1.5367804	-0.9863842

One shortcoming of this method is that if there are more than one categorical variables in the dataset, then we have to split the major dataset into sub datasets equal to the unique values of all the categorical datasets (a factorial problem), this could lead to intensive and time consuming computations, as well as class imbalance problems. Using an example: if our dataset contains the categorical variables Gender: Male or Female, and Smoker-status: Yes or No, then we have to split the major dataset into four sub datasets namely, the sub dataset that contains:

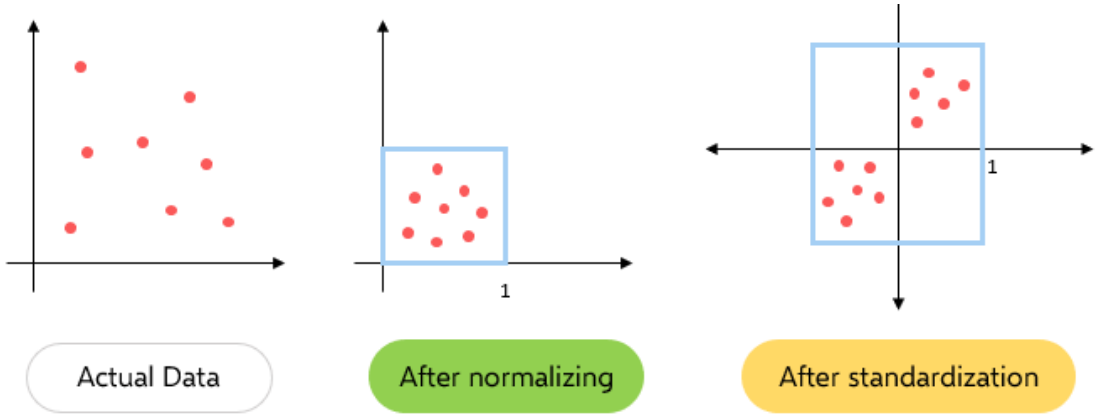
- Male smokers
- Male non smokers
- Female smokers
- Female non-smokers.

Thus for a dataset with four categorical variables, each having four unique values, we would create a total of 24 sub datasets!

There will always be a tradeoff between the two methods of dealing with the categorical variables, such that in the case of few categorical variables, where the issue of class imbalance might not arise, we can opt to use the splitting method, and in the case of many categorical variables, where splitting may lead to class imbalance problems, then we can choose the dummy coding or the one hot coding method for conversion of the dataset.

Scaling

As for the scaling issue, it is important to know that the distance measures are greatly affected by scales, and results obtained in instances where scaling was applied would differ greatly with the results where scaling was not applied. It is mostly advisable to apply scaling before the nearest neighbor search. Scaling can be done in two methods namely: min-max normalization and z-score standardization as shown below:



Distribution of Added noise

The simple linear regression model is given as:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i \; where \; i = 1, \dots, n$$

Where

$$y_i - is \; the \; response \; variable$$

$$\beta_0 - is \; the \; intercept$$

$$\beta_1 - is \; the \; slope \; of \; the \; regression \; line$$

$$x_i - is \; an \; independent \; variable$$

$$\varepsilon_i - is \; the \; random \; error \; term$$

We also know that:

$$E(\varepsilon_i) = 0$$

$$Var(\varepsilon_i) = \sigma^2$$

$$The \; error \; terms \; are \; uncorrelated, and \; independent, and \; follow \; a$$

$$normal \; distribution \; with \; mean \; 0 \; and \; variance \; \sigma^2$$

(I will stick to the x-y notation, where x is the independent variable and y is the dependent variable)

For a new test input x, we use the linear regression model to predict its y value, and then for every y value predicted, I will add an error term which is obtained by averaging the error terms of the nearest neighbor of our new x instance. Remember the nearest neighbors of the test instance are all obtained from the training set that was used to fit the linear regression model.

Therefore:

$$Let \; \widehat{y_i} \; be \; the \; predicted \; value \; of \; the \; new \; test \; instance \; x_{i.},$$

$$and \; let \; \theta_i be \; the \; error \; component \; that \; we \; desire \; to \; add \; to \; \widehat{y_i}$$

Assuming we obtain K nearest neighbors to the new test instance and we obtain their residuals from the linear regression model as shown below:

$$\varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4, \dots, \varepsilon_k$$

Then:

$$\theta_i = \frac{\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k}{k}$$

Thus for the new test instance, its predicted value will be given by:

$$\widehat{y_i} = \beta_0 + \beta_1 x_i + \frac{\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k}{k}$$

The desired error term has the following properties:

$$E(\theta_i) = E\left(\frac{\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k}{k}\right)$$

$$= \frac{1}{k} E(\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k)$$

$$= \frac{1}{k} \{E(\varepsilon_1) + E(\varepsilon_2) + E(\varepsilon_3) + \cdots + E(\varepsilon_k)\}$$

$$But\ E(\varepsilon_i) = 0, therefore$$

$$E(\theta_i) = \frac{1}{k} * 0 = 0$$

$$Var(\theta_i) = Var\left(\frac{\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k}{k}\right)$$

$$= \frac{1}{k^2} * Var(\varepsilon_1 + \varepsilon_2 + \varepsilon_3 + \cdots + \varepsilon_k)$$

$$= \frac{1}{k^2} \{Var(\varepsilon_1) + Var(\varepsilon_2) + Var(\varepsilon_3) + \cdots + Var(\varepsilon_k) + 2 * Covariance(\varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots, \varepsilon_k)\}$$

$$Due\ to\ the\ assumption\ that\ the\ error\ terms\ are\ uncorrelated\ and\ independent, their\ covariance\ vanishes$$

$$But\ Var(\varepsilon_i) = \sigma^2$$

$$= \frac{1}{k^2} * k\sigma^2$$

$$Thus: Var(\theta_i) = \frac{\sigma^2}{k}$$

We see that then the desired noise follows a Normal distribution with mean 0 and variance equal to $\frac{\sigma^2}{k}$

$$\theta_i \sim N(\mu = 0, \sigma^2 = \frac{\sigma^2}{k})$$

Experimental analysis

Case 1: Life expectancy Data

For the experimental analysis, I considered this first dataset from the World Health Organization (WHO) in conjunction with the United Nations. The dataset which is made freely available for download in the Kaggle repository has an aim of answering the question on factors affecting life expectancy by considering immunization factors, mortality factors, economic factors, social factors and other health related factors as well.

The original dataset includes all the countries, thus I had to filter and get Kenyan data. It is also important to note that in my data analysis, I discovered that the data exhibited multicollinearity, and thus I had to do away with several variables, while making sure that every category of factors i.e. social, immunization, mortality has at least one variable that ends up in the model.

The data is split into the training and testing set as follows:

- Training set – Data from 2002 to 2011
- Testing set – Data from 2012 to 2015.

After careful Exploratory Data Analysis, I ended up with the following variables:

1. Polio – This variable represents the total proportion of children immunized against Polio in that given year
2. Adult Mortality – This variable gives the adult mortality rates of that year.
3. Total expenditure – This variable reports the total expenditure of that particular year
4. Gross Domestic Product
5. Alcohol – This variable reports the total proportion of the population which consume alcohol products
6. Population – This variable gives the total number of people, as per the national census.
7. Life expectancy – This is the dependent variable which we wish to build a model and predict on. The life expectancy is given in years.

The fitted model is shown below:

```
Call:
lm(formula = Life.expectancy ~ Polio + Adult.Mortality + Total.expenditure +
    GDP + Alcohol + Population, data = k_train1)

Residuals:
        5         6         7         8         9        10        11
 0.001858  0.072929 -0.023476  0.005923 -0.053259 -0.053785 -0.046267
       12        13        14
-0.076267 -0.063436  0.235781

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        8.859e+01  1.194e+00  74.166 5.40e-06 ***
Polio             -1.303e-02  2.157e-03  -6.041  0.00910 **
Adult.Mortality   -6.772e-02  1.439e-03 -47.060 2.11e-05 ***
Total.expenditure -3.096e-01  1.606e-01  -1.928  0.14942
GDP                9.758e-04  2.668e-04   3.657  0.03531 *
Alcohol           -3.026e+00  4.586e-01  -6.598  0.00709 **
Population        -2.826e-08  5.367e-09  -5.265  0.01335 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1625 on 3 degrees of freedom
Multiple R-squared:  0.9994,    Adjusted R-squared:  0.9982
F-statistic: 812.3 on 6 and 3 DF,  p-value: 6.67e-05
```

The model appears to have almost all variables significant and the proportion of variability in Life expectancy which is explained by the model is 0.9982 implying that the model is a good fit to the data.

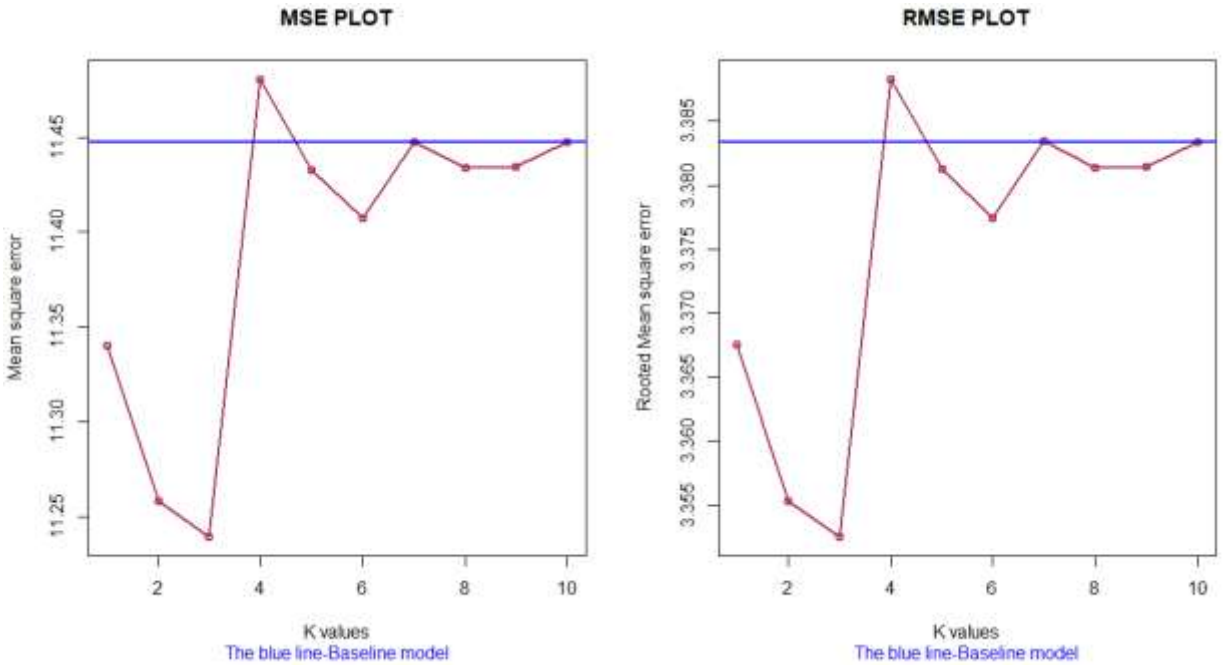
We will use the above model in predicting the life expectancy for the years 2012 to 2015, and then examine the error using the Mean squared error approach or the rooted mean squared error approach.

The error statistics are shown below:

Metric	*Value*
Mean Squared Error	11.44743
Rooted Mean Squared Error	3.383405

We further desire to examine the same metrics under the noise added regression models, and the visualizations below are useful.

In noise adding, since we have only 10 data points in the training dataset, we will try and fit 10 noise added models, using k equals to 1 neighbor, all the way to k equals 10 neighbors, all while recording useful statistics (MSE and RMSE).



As can be seen from the plots, where the maroon curves represent the MSE and RMSE of the noise added models, while the blue horizontal line is the MSE/RMSE of the regression model (without noise), majority of the noise added regression models perform better than the baseline model, with only one model performing poorly (that is: the k=4 neighbor noise added model) since it has a higher error metric as compared to the baseline model.

It is also important to note that the k=10 neighbor model has an error metric equivalent to the baseline model, since in the k=10, we are using all the data points to calculate the desired error term, and since an assumption of linear regression is that the sum of the error terms equals 0, then averaging 0 gives 0, which when added as noise is not different from the original baseline model.

Case 2: Swedish auto insurance dataset

For the second experimental analysis, I fetched the Swedish auto insurance dataset, which consists of only two variables:

- The number of claims
- Total payment for all the claims in Swedish Kronor for geographical zones in Sweden

Reference about the dataset can be found at: Swedish committee on Analysis of risk premium in motor insurance.

I split the dataset into a training set and a testing set as follows:

- Training set: 53 data points
- Testing set: 10 data points.

The fitted regression model is shown below:

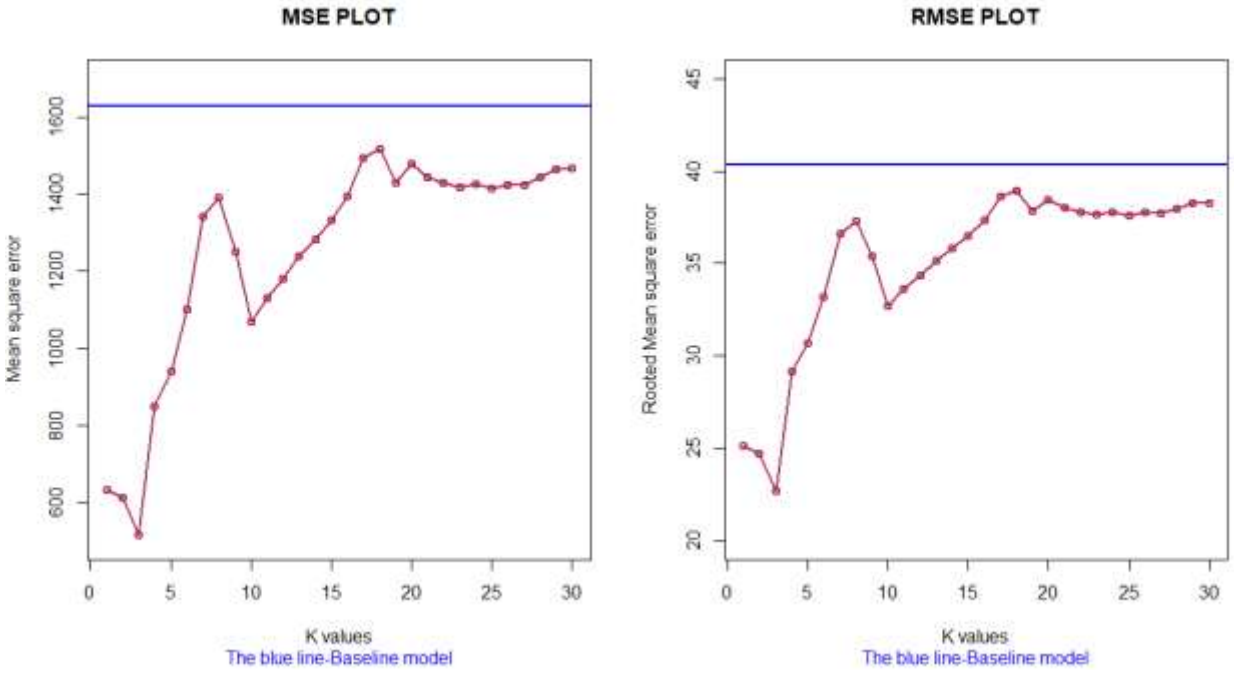
```
Call:
lm(formula = total_payment ~ claims_no, data = dd_train)

Residuals:
    Min      1Q  Median      3Q     Max
-89.463 -23.883   0.401  22.676  81.093

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  22.8205     6.5351   3.492 0.000999 ***
claims_no     3.4157     0.1998  17.094  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 35.15 on 51 degrees of freedom
Multiple R-squared:  0.8514,    Adjusted R-squared:  0.8485
F-statistic: 292.2 on 1 and 51 DF,  p-value: < 2.2e-16
```

It can be seen that both the intercept and the claims number variable are significant in explaining the total payment on claims. The R squared value reported 0.8485 is higher implying that approximately 86% of the variation in total payments is accounted for by the model.

The error statistics for the pure regression model are shown below:

Metric	*Value*
Mean Squared Error (MSE)	1629.122
Rooted Mean Squared Error (RMSE)	40.36238

For the case of the noise added regression models, I fitted a total of 30 models, for the k equals 1 neighbor to the k equals 30 neighbours models, and the error statistics are visualized as follows:



A can be shown by the visualization above, all the noise added regression models performed far better than the pure (without noise) model, with some even performing 50% much better in predictions.

It is also noticeable that as more noise is added, the error tends to rise, thus favorable noise comes from around 4 to 7 neighbors.

With these two examples of experimentation analysis, we can have more confidence in the noise added models.

Conclusion

Since it is evident that the noise adding leads to improvement in the predictive sense of the regression models, then it is valuable that they should be preferred in actuarial work which places an importance on accuracy of predictions.

It is also noteworthy that the noise adding does not increase any complexity in the nature of simple and multiple linear regression model, rather the parsimony of the regression model is not tampered with since the noise added is in fact extracted from the regression model and not from an outward source. Also the noise added models do not lead to an increased number of parameters to be estimated, rather inference is only based on the pure regression model thus preserving the simplicity and assumptions of the regression model.

It is also noteworthy that the noise added regression model thrive both when the size of the data is small (i.e. fewer data points for analysis), as well as for the case of big data.

Thus this model may be incorporated into actuarial works involving regression analysis in a predictive sense.

Miscellaneous

Choosing the Value of K for the nearest neighbor algorithm

Since the nearest neighbor algorithm relies on only one parameter k (the number of neighbors), then the choice of k is crucial, and different values of k often yield different results. But in our case, it is mostly advisable to choose a large enough k, depending on the dataset since for mostl of the analyses I have conducted using the noise added models, a value of k=5 and above always yields improved results.

Another solution to combat this is cross training or corss validation, which implies having three sets of data, the training set, the validation set and the testing set. The validation set can be sued to try several values of k before deciding which is the best.

One can also employ model averaging. Model averaging implies combining models, e.g. we may decide to use k=4, 5, 6, 7 and instead of choosing the best model, we average the results of all the four models, in order to arrive at the final results. This will help since the averaged results will combine the effect of all the models, and might thus be favourable.

The case for Big Data regression:

In the case where actuaries eed to run noise adding effect in linear regression on huge volumes of data, it should be noted that the K nearest neighbour algorithm might tend to be computationally expensive to use, since assume given a training dataset of 100,000 data points (rows), and a testing dataset of 10,000 data points, then when running the k nearest neighbour algorithm, the algorithm calculates the distance of each test set datapoint from the training point, and this might lead to 100,000 * 10,000 = 1,000,000,000 calculations, and so on.

But to counter this challenge, we could employ the use of the **Learning Vector Quantization** algorithm, which is a lot like the k nearest neighbor algorithm in that:

The nearest neighbor algorithm relies on the entire training dataset in order to learn which makes it computationally intensive, while the LVQ algorithm addresses this problem by learning a much smaller subset (only a portion of the training dataset) of patterns that best represent the training data, and by doing so becomes less computationally intensive.

References

Triguero, I., García-Gil, D., Maillo, J., Luengo, J., García, S., & Herrera, F. (2019). Transforming big data into smart data: An insight on the use of the k-nearest neighbors algorithm to obtain quality data. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *9*(2), e1289.

Weisberg, S. (2005). *Applied linear regression* (Vol. 528). John Wiley & Sons.

Frees, E. W. (2009). *Regression modeling with actuarial and financial applications*. Cambridge University Press.

Mevik, B. H., & Cederkvist, H. R. (2004). Mean squared error of prediction (MSEP) estimates for principal component regression (PCR) and partial least squares regression (PLSR). *Journal of Chemometrics*, *18*(9), 422-429.

Life expectancy dataset: https://www.kaggle.com/kumarajarshi/life-expectancy-who

Swedish auto insurance dataset: https://www.kaggle.com/ahmedjaved701/swedish-auto-insurance-dataset