Machine Learning –Regression

Thesis · January 2022

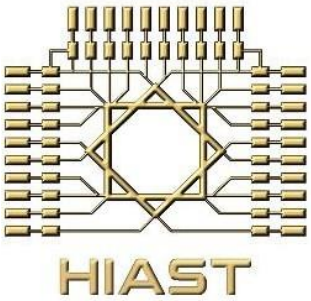
DOI: 10.13140/RG.2.2.35768.67842

2 authors:

Haidara Saleh
Higher Institute for Applied Sciences and Technology
1 PUBLICATION **2** CITATIONS
SEE PROFILE

Jamil Antone Layous
Higher Institute for Applied Sciences and Technology
10 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Syrian Arab Republic
Higher Institute for Applied Sciences and Technology
Department of Electronic and Mechanical Systems
Mechatronics Engineering - fourth year



Machine Learning – Regression

By:

Haidara SALEH

Scientific Supervisor: M.Sc. Jamil LAYOUS

Organizer Supervisor: Dr. Adel ALKAFRI

Linguistic Supervisor: Mr. Fahmi ALAMMAREEN

Damascus 2021-2022

Contents

List of figures

Acknowledgments

I thank M.Sc. Jamil LAYOUS for his guidance through each stage of the process …
I thank Mr. Fahmi ALAMAREEN for his hard work in thoroughly reading and correcting this research linguistically ...
I thank Dr. Adel ALKAFRI for his notes and technical advice ...

Abstract

The goal of a regression model is to build a mathematical equation that defines y (the outcome variable) as a function of one or multiple predictor variables (x). Next, this equation can be used to predict the outcome (y) based on new values of the predictor variables (x). This research tackles the main concepts considering Regression analysis as a statistical process consisting of a set of machine learning methods including data splitting and regularization, we start by introducing the meaning of supervised and unsupervised learning and then we develop motivation for algorithms used in data splitting. Regression methods are then discussed with fair length focusing on linear regression. We conclude the research with an application of a real-life regression problem.

1 Introduction to Machine learning: what and why?

Machine learning is one of today's most rapidly growing technical fields and one of the fastest-growing areas of computer science, with far-reaching applications. It's the field of study that gives computer algorithms the ability to learn without being explicitly programmed.

It Lies at the intersection of computer science and statistics, and at the core of artificial intelligence. it addresses the question of how to build computers that improve automatically through experience.

Machine learning is usually divided into two main types depending on the **predictive** or **supervised learning** approach, and the **descriptive** or **unsupervised learning** approach.

1.1 SUPERVISED MACHINE LEARNING

Supervised learning is the major practical machine learning used nowadays. A supervised scenario can be described by the concept of a teacher or supervisor whose main task is to provide the agent with a precise measure of its error (directly comparable with output values). Starting from this information, the agent can correct its parameters so as to reduce the magnitude of a global loss function after each iteration, in other words, it is where you have input variables (X) and an output variable (Y) and you choose an algorithm to learn the mapping function from the input to the output: Y = f(X) and this function is provided by a **dataset** which is a collection of **labeled examples** $\{(\mathbf{x}i, yi)\}^{N}_{i=1}$, Each element $\mathbf{x}i$ among N is called a **feature vector**. A feature vector is a vector in which each dimension $i = 1, \ldots, D$ contains a value that describes the example somehow.

Learning stops when the algorithm achieves an acceptable level of performance but it doesn't end here because our goal here is to approximate the mapping function so well in order to train a system that must also work with samples that have never been seen before. So, it's necessary to allow the model to develop a generalization ability and avoid a common problem called **overfitting** which we will discuss later in this research.

Supervised learning problems can be further grouped into regression (which we will be focusing on in this paper) and classification problems.

- **Classification**: A Classification problem is when the output variable is a category.
- **Regression**: A regression problem is a problem of predicting a real-valued label and the response variable is a continuous value.

1.2 Unsupervised Machine learning

This approach is based on the absence of any supervisor and therefore of absolute error measures. So, it's called unsupervised learning because unlike supervised learning mentioned above there is no correct answer and there is no teacher. Algorithms are left to their own devices to discover and present interesting data structures. The dataset here is a collection of **unlabeled examples** $D = \{\mathbf{x}i\}^{N}_{i=1}$ where again **x** is a feature vector, and the goal of an **unsupervised learning algorithm** is to create a **model** that takes a feature vector **x** as input and either transforms it into another vector or into a value that can be used to solve a practical problem, so we are trying to find "interesting patterns" in the data. This is sometimes called **knowledge discovery**. This is a much less well-defined problem, since we are not told what kinds of patterns to look for, and there is no obvious error metric to use (unlike supervised learning, where we can compare our prediction of ***y*** for a given **x** to the observed value).
Unsupervised learning problems can be further grouped into clustering and association problems.

- **Clustering**: A Clustering problem is when you want to discover natural grouping in data, and figure 1 explains clustering with a bidimensional dataset split into four natural clusters.

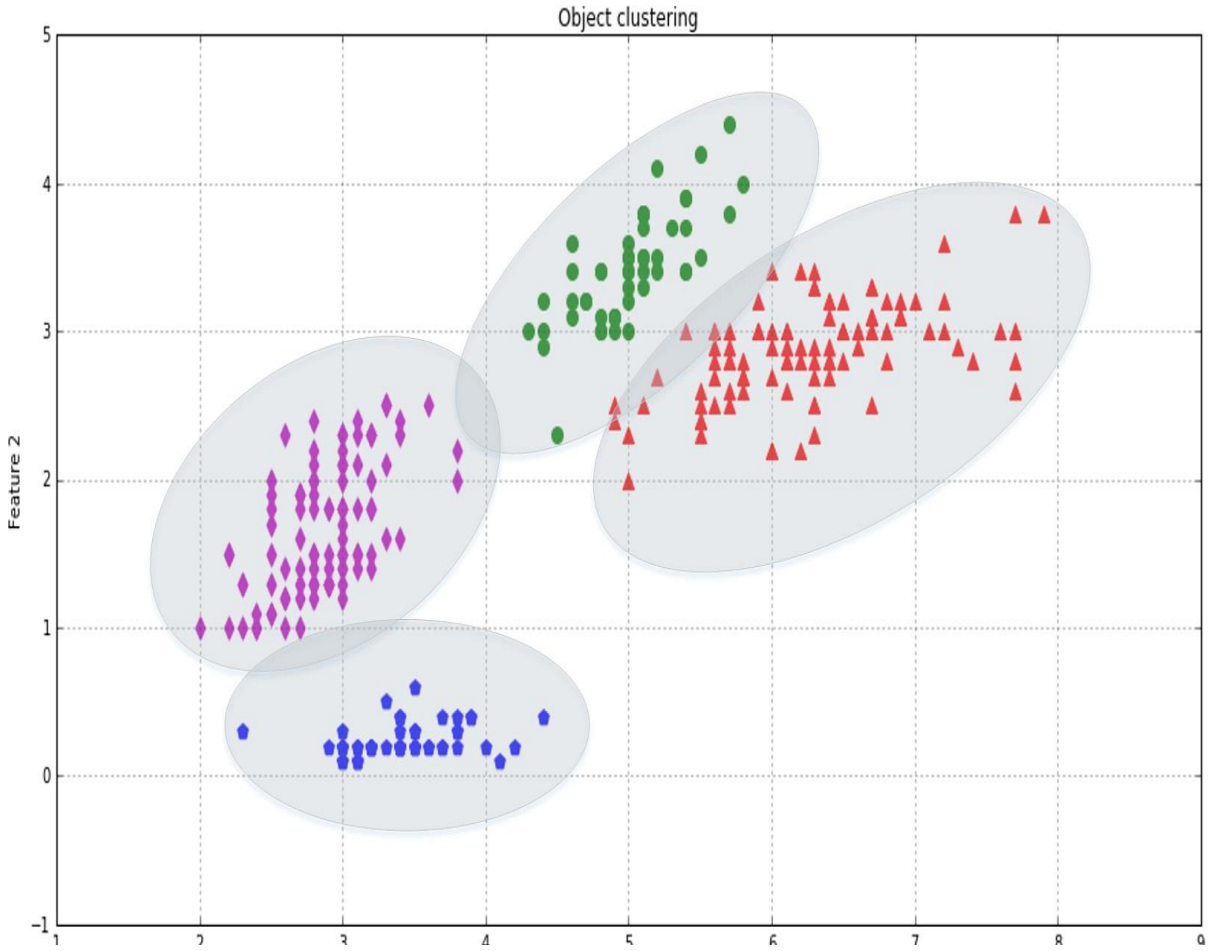


Figure 1: Example of clustering with a bidimensional dataset split into four natural clusters [1]

- **Association**: It is used to identify new and interesting insights, associations, and relationships between different objects in the data set and frequent patterns in transactional data. So, an association learning problem is where you want to discover rules that describe large portions of your data, such as people that buy item A also tend to buy item B, and figure 2 explains how an association learning problem can be solved.

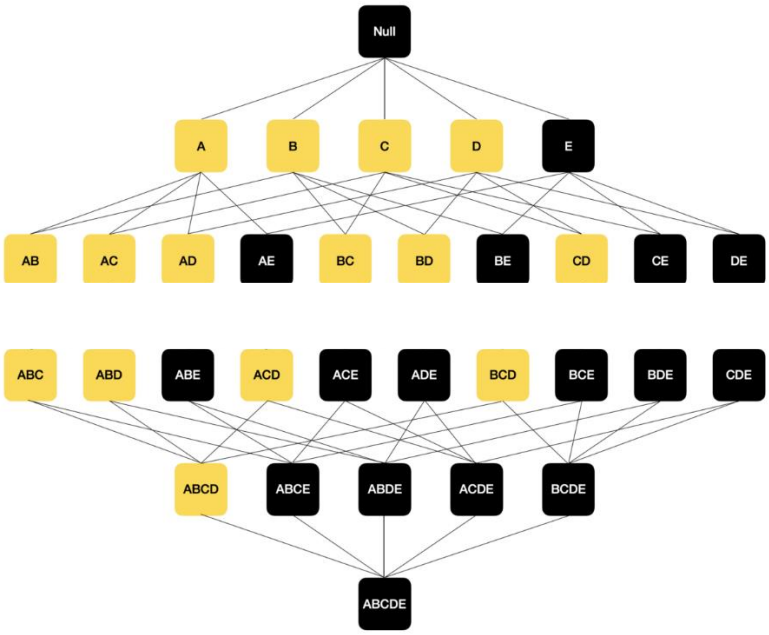


Figure 2: Example of association learning

2 Data Splitting

In machine learning, one of the main requirements is to build computational models with high prediction and generalization capabilities [2]. In the case of supervised learning, a computational model is trained to predict outputs of an unknown target function. The target function is represented by a finite training dataset T of examples of inputs and the corresponding desired outputs:

T = { [~x1 , ~d1], . . . , [~xn , ~dn] }, where n > 0 is the number of ordered pairs of input/output samples (patterns).

The final model that comes at the end of the training process should predict correct outputs for the input samples from T, but it should also be able to generalize well to previously unseen data. Poor generalization can be characterized by over-training. If the model over-trains, it just memorizes the training examples and it will not be able to give correct outputs also for patterns that were not in the

training dataset. These two crucial demands (good prediction on T and good generalization) are conflicting and are also known as the Bias and Variance dilemma [3].
Generally, in order to train the model, this data is split into different subsets (sub-datasets), these subsets are used for model training, testing, and validation. In terms of model training, model parameters (e.g. node weights) are derived, and the learning process is controlled externally by the so-called, hyper-parameters. While the training subset is used to fit the model parameters, the validation takes the role to tune the model hyper-parameter and giving an estimate of the skill level of the model. also, the testing dataset is used for estimating the final tuned model skill level [4].
However, estimating the model performance on the same training data will lead to biased estimates, thus both, validation and testing datasets, should be held out from the original dataset. Commonly, these subsets are chosen and divided either split-by-hand or randomly [5], where the earliest requires extensive effort, the latter risks the loss of good feature representation. In real-world datasets, it is very exhaustive to prepare training, validation, and testing sub-datasets that comprise a good representation of the features and categories within the dataset. Interestingly, machine learning models perform excellently on split by hand-curated datasets, which is not the case in real-world data. The most common practice among data scientists is to split the dataset randomly with repetitive iterations until approaching a rational performance of their machine models.
There are many data splitting methods reported and used in supervised learning and these methods can be roughly categorized into three different types:

1. Randomly, selecting a proportion of samples for training and retaining the remaining ones (holding out) as a validation set. Usually, we repeat this process many times and the final estimation of the model performance is the average performance on validation sets of all the iterations; the best-known method used for this type of repartitioning of the data is probably the bootstrap as proposed by Efron et al. [6].

2. cross-validation (CV), which is a data resampling method to assess the generalization ability of predictive models and to prevent overfitting, and has two types which are hold-out cross-validation (early stopping) and k-fold cross-validation.

3. Based on the distribution of the data, systematically selecting a given number of the most representative samples from the datasets and using the

remaining samples for validation. Kennard-Stone algorithm (K-S) [7] is a good example of such a method.

Now we will take a look at three algorithms. Each one of them describes one of the previous categories:

2.1 Leave-One-Out Bootstrap

The bootstrap is a data resampling method for estimating the statistical parameters of an unknown distribution such as mean, median, variance, confidence interval, etc. [8]. It has been later proved to be a good resampling method for model selection [9]. Given *n* samples available in the data, bootstrap randomly chose n' *samples.* These samples are used as the training set and the unselected samples are used as the validation set. The ratio of the samples in the training and validation set is variable and on average 63.2% samples would be used as a training set and 36.8% samples would be used as a validation set. then the bootstrapping begins:
Walking through it step by step, the bootstrap method works like this:
After dividing our given dataset of size n to the training dataset of size n' and the validation dataset of size $(n - n')$.

1. For $j = 1, \cdots, b$: b = bootstrap bags:
   We draw one single instance from this dataset and assign it to the jth bootstrap sample set (often called bag) with replacement; i.e., the same sample can be chosen multiple times and some samples may not appear at all. We repeat this step until our bootstrap sample has a size of n'' – which is usually 60% of the original training dataset size.
2. We fit a model to each of the b bootstrap sample bags and compute the resubstitution accuracy.
3. We compute the model accuracy as the average over the b accuracy estimates as in Equation 1.

$$\frac{1}{b}\sum_{j=1}^{b}\frac{1}{n''}\sum_{i=1}^{n''}[\,1 - L(\hat{y}i, yi)\,] \quad where\ L\ is:\ (y(i) - \hat{y}(i))^2 \qquad (1)$$

2.2 K-fold cross-validation

It is probably the most common technique for model evaluation and model selection in machine learning practice. The k-fold cross-validation [1] uses a combination of more tests to gain a stable estimate of the model error. It is useful if not enough data for the hold-out cross-validation is available.
the available learning set is partitioned into k disjoint subsets of approximately equal size. Here, "fold" refers to the number of resulting subsets. This partitioning is performed by randomly sampling cases from the learning set without replacement. The model is trained using k -1 subsets, which, together, represent the training set. Then, the model is applied to the remaining subset, which is denoted as the validation set, and the performance is measured. This procedure is repeated until each of the k subsets has served as a validation set and is done in such a way that no two test sets overlap. The average of the k performance measurements on the k validation sets is the cross-validated performance.
Algorithm 1 describes this method in more detail, and figure 3 illustrates the k-fold cross-validation procedure.

Algorithm 1 K-fold cross-validation
1. Input: dataset $\boldsymbol{T}$*, number of folds* $\boldsymbol{k}$*, performance function error, computational models* $\boldsymbol{L_1, \cdots, L_m, m \geq 1}$
2. Divide T into k disjoint subsets $\boldsymbol{T_1, \cdots, T_k}$ *of the same size.*
3. For i = 1, · · ·, k:
$\boldsymbol{Tv \leftarrow T_i, T_{tr} \leftarrow \{T \setminus T_i\}}$**.**
3.1. For j = 1, · · ·, m:
Train model $\boldsymbol{L_j}$ *on* $\boldsymbol{T_{tr}}$ *and periodically use Tv to assess the model performance:*
$\boldsymbol{E^j_v(i)}$*=error (*$\boldsymbol{L_j(T_v)}$*).*
Stop training, when a stop-criterion based on $\boldsymbol{E^j_v(i)}$ *is satisfied.*
4. For j = 1, · · ·, m, evaluate the performance of the models by: $\boldsymbol{E^j_v = \frac{1}{k}\sum_{i=1}^{k} E^j_v(i)}$
As an Error measure function, we could use Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}}$$

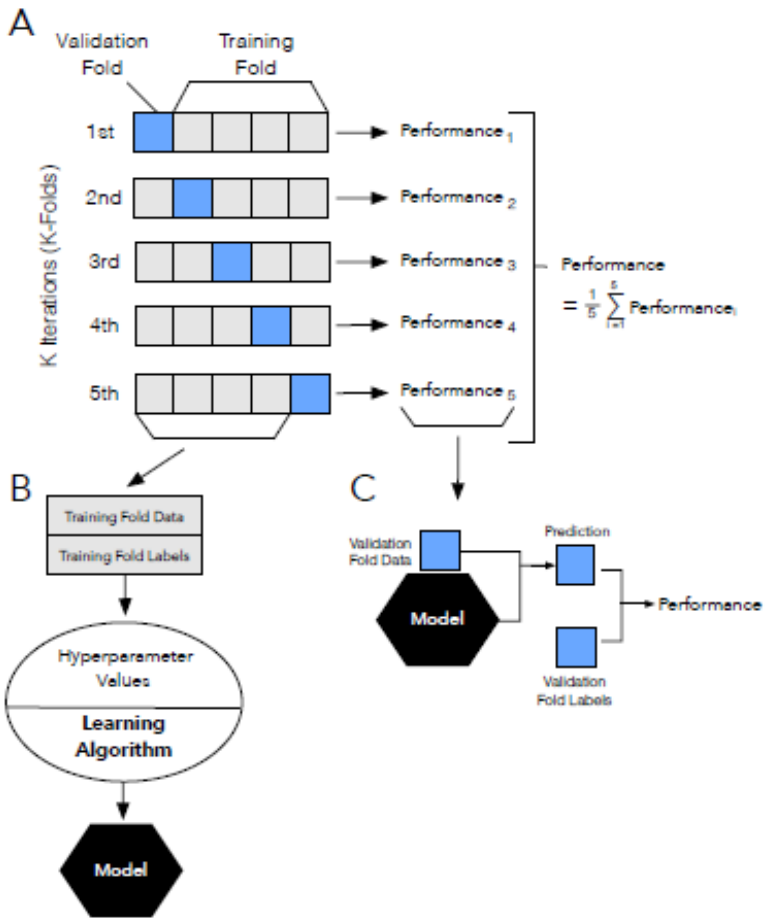


Figure 3: Illustration of the k-fold cross-validation procedure

2.3 Kennard-Stone Sampling

The K-S algorithm [10], also known as the computer-aided design of experiment (CADEX) algorithm, is designed to select the most representative samples from a given dataset. K-S employed a stepwise procedure. So, it starts by finding two samples that have the largest Euclidean distance; these are the two samples furthest apart, and ranking them as the most representative.

Then in each following step, the remaining samples having the greatest distance from the already selected samples are chosen and added to the bottom of the previous rank list. This procedure is repeated until a predefined number of samples had been chosen and ranked. These selected samples are usually used as the training set since a representative dataset is crucial for training a good model and the remaining samples are used as the validation set. Unlike CV and bootstrap, there is only one split of training and validation set in the K-S algorithm. The result is that, for a given fraction of the data to be parsed into the calibration set (training

set), samples are selected to smoothly fill the data space. An example is shown in Figure 4 for a synthetic data set that has 1000 samples and 2 variables with a multivariate normal distribution. In this case, 66% of the samples were placed into the calibration set. As expected, the test set is interior to the exterior points of the calibration set. This splitting, therefore, avoids extrapolation when a model is applied to the test set. We can see an illustration of how Samples are selected using the Kennard-Stone algorithm: calibration set (blue circles) and test set (red squares) in figure 4.

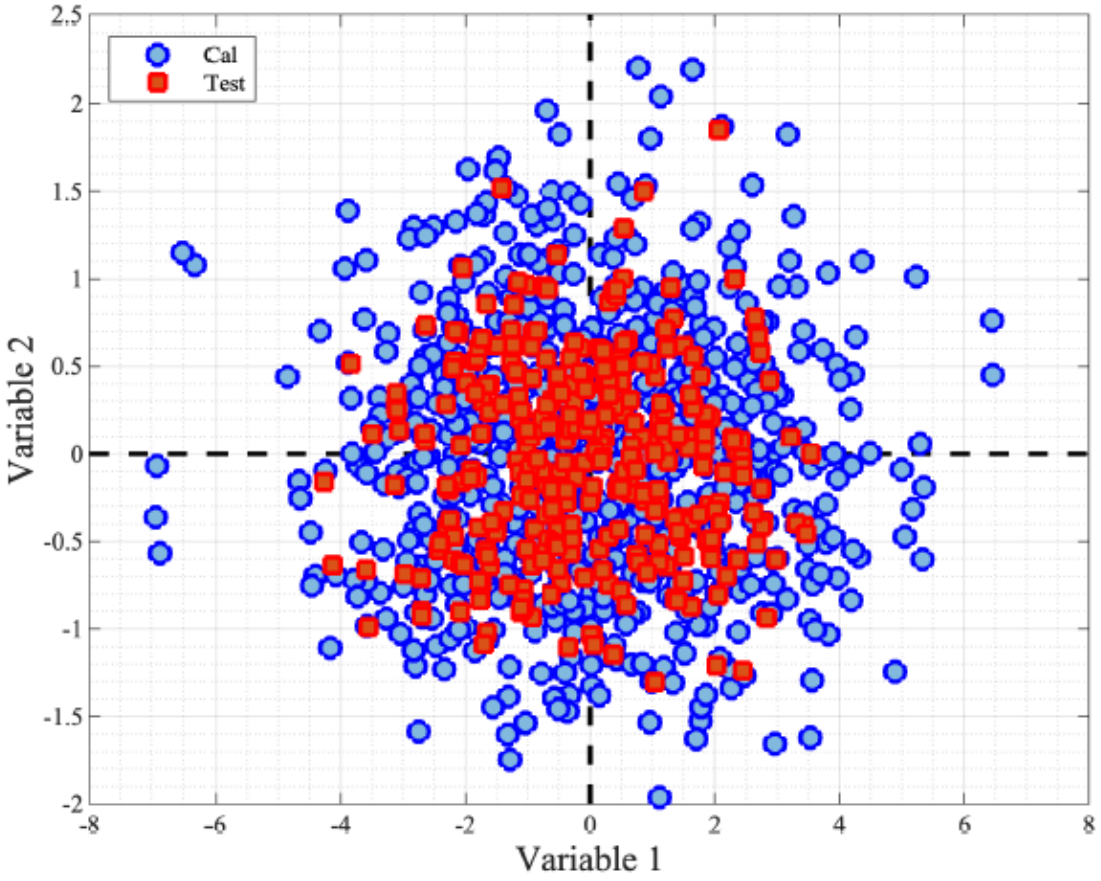


Figure 4: Illustration of how Samples are selected using the Kennard-Stone algorithm [11]

3 Regression Methods

Regression is a set of techniques for estimating relationships, once we have acquired data with multiple variables, we try to figure out how the variables are related. For example, we could ask for the relationship between people's weights

and heights, or study time and test scores, or, we can explain how many transistors the semiconductor industry can pack into a circuit over time (Moore's law, shown in Figure 5).

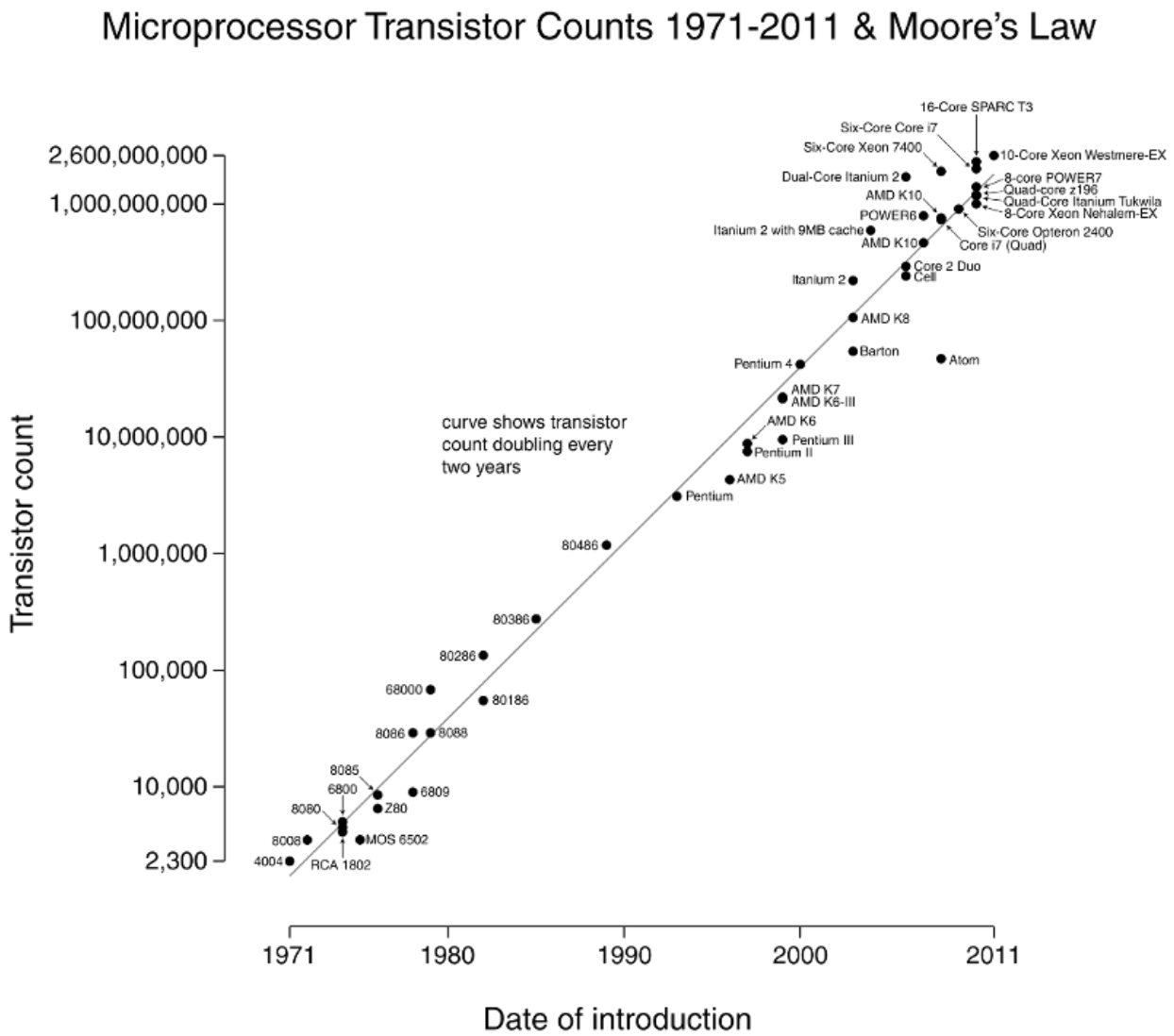


Figure 5: Illustration of Moore's law over time [12]

3.1 Linear regression

Linear regression is a popular regression learning algorithm, and probably the most basic type of regression and is commonly used for predictive analysis projects. In fact, when you are working with a single predictor (variable), we call it simple linear regression, and if there are multiple predictor variables, we call it

multiple linear regression. Simply put, linear regression uses linear predictor functions whose values are estimated from the data in the model.

Linear models are the simplest parametric methods and always deserve the right attention, because many problems, even intrinsically non-linear ones, can be easily solved with these models. As discussed previously, a *regression* is a prediction where the target is continuous and it has several applications, so it's important to understand how a linear model can fit the data, what its strengths and weaknesses are, and when it's preferable to pick an alternative.

3.1.1 Simple Linear regression

The word ‘simple’ means that there is a single independent variable, but the word linear does not have a meaning that would seem to be self-evident. Specifically, it does not mean that the relationship between the two variables can be displayed graphically as a straight line. Rather, it means that the model is linear in the parameters. The basic model is

$$Y=ß0+ßlX+\varepsilon \qquad (2)$$

in which Y and X denote the dependent and independent variables, respectively, and ßo and ßl are parameters that must be estimated. The symbol ε represents the error term. This does not mean that a mistake is being made; it is simply a symbol used to indicate the absence of an exact relationship between X and Y. The reader will recognize that, except for e, Eq. (2) is in the general form of the equation for a straight line. That is, ßl is the slope and ßo is the Y-intercept.

3.1.2 Multiple Linear regression

Most applications of regression analysis involve the use of more than one regressor. The model for multiple linear regression is:

$$Y=ß0+ßlX1+ ß2X2+ \cdots + ßmXm+\varepsilon \qquad (3)$$

with the corresponding prediction equation:

$$\widehat{Y} = \widehat{ß}0 + \widehat{ß}1X1+\widehat{ß}2X2 + \cdots + \widehat{ß}mXm \qquad (4)$$

As in Eq (3), m will hereinafter denote the number of regressors. Multiple regression is, unfortunately, sometimes referred to as multivariate regression to indicate that there is more than one regressor. The two terms are not synonymous. however. The word multivariate must be reserved for the case where there is more than one dependent variable. Although multivariate regression is indeed a field of study. Even though we may appropriately regard multiple regression as an extension of simple regression, there are some questions that the user of multiple

regression must address that are not encountered in simple regression. In particular, if data are available on. say, k variables that might seem to be related to the dependent variable, should all k variables be used? If not, which ones should be used? What is gained, if anything, by having m < k? Can we use scatter plots to determine which independent variables to include in the model? Can possible transformations of the regressors be determined simply by examining such scatter plots? Should alternatives to least squares be used under certain conditions? If so, under what conditions should they be used, and which ones should be considered? Specifically, should least-squares still be used when there are high correlations between the regressors?

3.1.2.1 Problem Statement

We have a collection of labeled examples $\{(\mathbf{x}i, \boldsymbol{y}i)\}_{i=1}^{N}$, where N is the size of the collection, $\mathbf{x}i$ is the D-dimensional feature vector of example $i = 1, \ldots, N$, yi is a real valued target and every feature xi , j = 1, . . . , D, is also a real number.

We want to build a model $fw,b(x)$ as a linear combination of features of example x:

$$fw,b(x) = wx + b \qquad (5)$$

where w is a D-dimensional vector of parameters and b is a real number. The notation fw,b means that the model f is parametrized by two values: w and b.
We will use the model to predict the unknown y for a given x. Two models parametrized by two different pairs (w, b) will likely produce two different predictions when applied to the same example. We want to find the optimal values (w∗, b∗). The optimal values of parameters define the model that makes the most accurate predictions.
The hyperplane in linear regression is chosen to be as close to all training examples as possible. We can see why this latter requirement is essential by looking at the illustration in Figure 6, It displays the regression line (in red) for one-dimensional examples (blue dots). We can use this line to predict the value of the target y_{new} for a new unlabeled input example x_{new}. If our examples are D-dimensional feature vectors (for $D > 1$), the only difference with the one-dimensional case is that the regression model is not a line but a plane (for two dimensions) or a hyperplane (for $D > 2$).

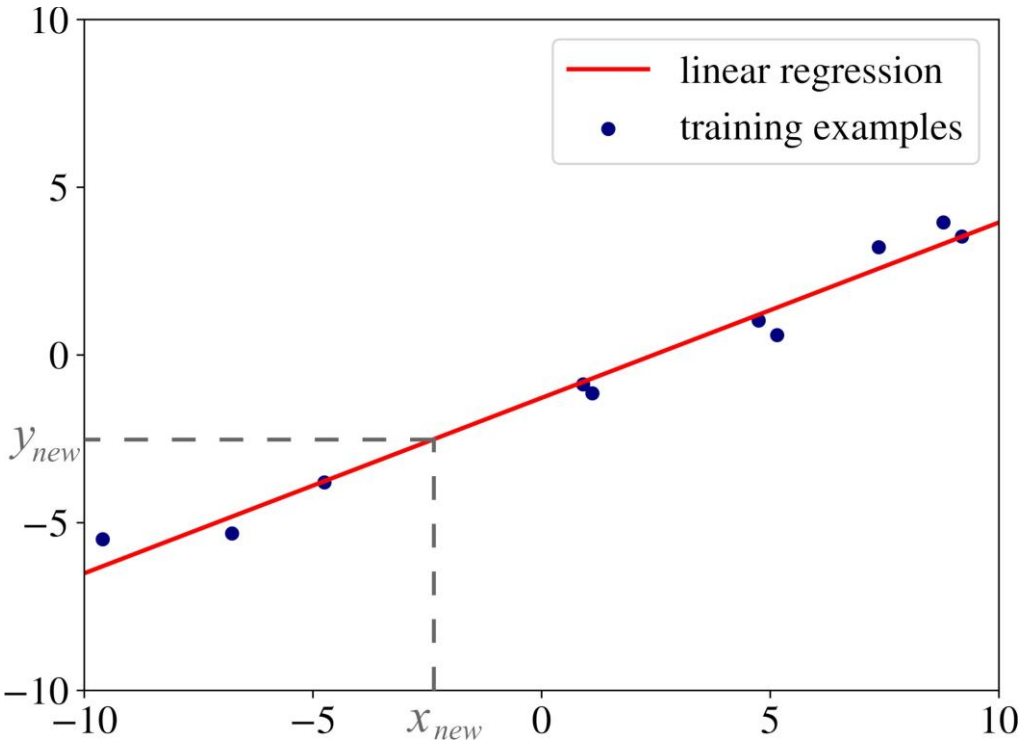


Figure 6: Linear Regression for one-dimensional examples

3.1.2.2 Solution

To get this latter requirement satisfied, the optimization procedure which we use to find the optimal values for $\mathbf{w}*$ and b^* tries to minimize the following expression:

$$\frac{1}{N}\sum_{i=1}^{N}[\, fw, b(xi) - yi\,]^2 \qquad (6)$$

In mathematics [13], the expression we minimize or maximize is called an objective function, or, simply, an objective. The expression $(f_{\mathbf{w},b}(\mathrm{x}_i)$ - $y_i)^2$ in the above objective is called the **loss function**. It's a measure of penalty for misclassification of example *i*. This particular choice of the loss function is called **squared error loss**. All model-based learning algorithms have a loss function and what we do to find the best model trying to minimize the objective known as the **cost function**. In linear regression, the cost function is given by the average loss,

also called the **empirical risk**. The average loss, or empirical risk, for a model, is the average of all penalties obtained by applying the model to the training data.

3.2 Polynomial Regression

Polynomial regression is a technique based on a trick that allows the use of linear models even when the dataset has strong non-linearities. The idea is to add some extra variables computed from the existing ones and using (in this case) only polynomial combinations:

$$\tilde{y}_k = \alpha_0 + \sum_{i=1}^{m} \alpha_i \bar{x}_k^{(i)} + \sum_{j=m+1}^{k} \alpha_j f_{p_j}\left(\bar{x}_k^{(1)}, \bar{x}_k^{(2)}, \ldots, \bar{x}_k^{(m)}\right) \qquad (7)$$

In the previous expression, every $f_{Pj}(\bullet)$ is a polynomial function of a single feature. For example, with two variables, it's possible to extend to a second-degree problem by transforming the initial vector (whose dimension is equal to m) into another one with higher dimensionality (whose dimension is $k > m$):

$$\bar{x} = (x_1, x_2) \rightarrow \bar{x}_t = (x_1, x_2, x_1^2, x_2^2, x_1 x_2) \qquad (8)$$

3.3 Regularization

Regularization is an umbrella-term that encompasses methods that force the learning algorithm to build a less complex model. In practice, that often leads to slightly higher bias but significantly reduces the variance. This problem is known in the literature as the **bias-variance tradeoff**.
With regularization, whether we are speaking about mathematics, statistics, or machine learning, we are essentially talking about a process of adding additional information in order to solve a problem. The term regularization has been described as an abstract concept of management of complex systems (according to a set of rules or accepted concepts). These rules will define how one can add or modify values in order to satisfy a requirement or solve a problem.

there are certain indicators to observe that should cause you to consider regularization, for example:

1- If your data contains a high variable count
2- If there is a low ratio of the number of observations to the number of variables in your data.

most popular statistical regularization methods may include the following:

3.3.1 Ridge

Ridge regression is a statistical technique that is used when analyzing regression data or models that suffer from a condition known as multicollinearity* or ill-conditioning (**Multicollinearity** is a condition within statistics in which a predictor (variable) in multiple regression models can be linearly predicted from the others with a significant accuracy). When multicollinearity occurs, estimates may be unbiased but their variances are usually large and far from the true value. This technique adds a degree of bias to the regression estimates to reduce standard errors (to produce more dependable estimates).

3.3.2 Lasso

The Least absolute shrinkage and selection operator (Lasso) is a statistical technique that performs both variable selection and regularization in an effort to enhance prediction accuracies within a model. The process of choosing or selecting variables within statistical model results, obviously, in reducing the number of variables, which is also referred to as variable shrinkage.

3.3.3 ElasticNet

The last alternative is called **ElasticNet** and combines both Lasso and Ridge into a single model with two penalty factors: one proportional to *Lasso* norm and the other to *Ridge* norm. In this way, the resulting model will be sparse like a pure Lasso, but with the same regularization ability as provided by Ridge. The resulting loss function is as follows:

$$L = \left\|Y - X\bar{\theta}\right\|_2^2 + \alpha\beta\left\|\bar{\theta}\right\|_1 + \frac{\alpha(1-\beta)}{2}\left\|\bar{\theta}\right\|_2^2 \qquad (9)$$

The **ElasticNet** class provides an implementation where the alpha parameter works in conjunction with l1_ratio (beta in the formula**)**. The main peculiarity of **ElasticNet** is avoiding a selective exclusion of correlated features, thanks to the balanced action of the *Ridge* and *Lasso* norms.
the performance of **ElasticNet** is superior to both **Ridge** and **Lasso** because it combines the shrinkage effect of the former and the feature selection of the latter. However, as the interaction of the two penalties is more complex to predict, I always suggest performing a grid search with a large range of parameters. Whenever necessary, it's possible to repeat the process by *zooming* into the range

containing the parameters previously selected, so to find out possibly the most performing combination.

4 Regression Application in House Price Prediction

The real estate industry is one of the leading research projects focusing on modern economics, for its significant implications on relevant industries and fields such as construction and investment. Constructing a realistic model to predict the price of real estate has been a challenging topic, but now with the help of modern machine learning techniques and with today's powerful possessing capability, finding the optimal solution became easily in our hands and with high accuracy.
In this section, we tried to apply a supervised machine learning approach to solve the problem discussed above, and we chose to use a multiple linear regression algorithm to train a model based on a training dataset that has many features of a typical house in the United States of America as the number of bedrooms, the number of bathrooms and the house area in square feet … as showed in table 1.

price	bedrooms	bathrooms	sqft_living	floors	sqft_basement	yr_built	yr_renovated	zipcode
221900	3	1	1180	1	0	1955	0	98178
538000	3	2	2570	2	400	1951	1991	98125
180000	2	1	770	1	0	1933	0	98028

Table 1: some features used in the training dataset

We used **TuriCreate** library which is a python library that simplifies the development of custom machine learning models, and we got our model trained with the help of **Colab** which is a product from Google Research that allows anybody to write and execute arbitrary python code through the browser and is especially well suited to machine learning, data analysis, and education.
In order to scale to **big data** (data that is so large, fast, or complex that it's difficult or impossible to process using traditional methods) we used **Sframe** from **TuriCreate** which means scalable data frame. A tabular, column-mutable dataframe object (data structure that organizes data into a 2-dimensional table of rows and columns). In our model we split the available data which contains information from 21614 houses randomly to 80% for training dataset and 20% for testing dataset and for the first stage we used a simple linear regression approach to train the model with the house space as our only variable in the input and its price as the output, so we got the following model with the coefficients:
(ß0,ß1) = `(-44127.305424643, 281.14227269104)` and after evaluating we got a Root Mean Square Error (RMSE) of 253933.857151. The linear model with its data set is shown in figure 7 where the y axis represents the price of the

house with 100000$ as a unit, and the x axis represents the house space in square feet, and we can see the used code below the following figure.

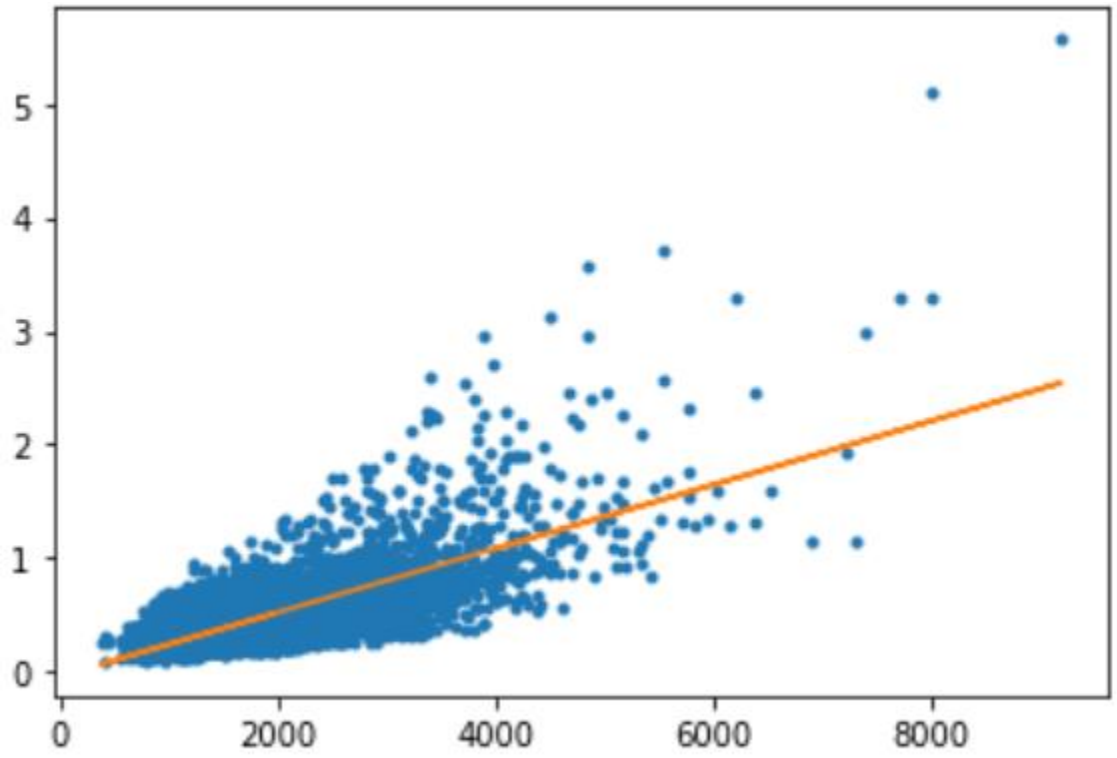


Figure 7: Illustration of the linear model with the data set

The used code in stage 1:

```
import turicreate as tc
from turicreate import SFrame
sales = tc.SFrame('/content/drive/MyDrive/home_data.sframe/home_data.sframe')
;
train_data,test_data = sales.random_split(.8,0)
model = tc.linear_regression.create(train_data,target = 'price',features = ['
sqft_living'])
print(model.evaluate(test_data))
plt.plot(test_data['sqft_living'],test_data['price'],'.',test_data['sqft_livi
ng'],model.predict(test_data),'-') // for drawing the linear model with the
data set
```

And for a better declaration of the house prices with its spaces we can see the first one thousand samples of our data set showed in figure 8.

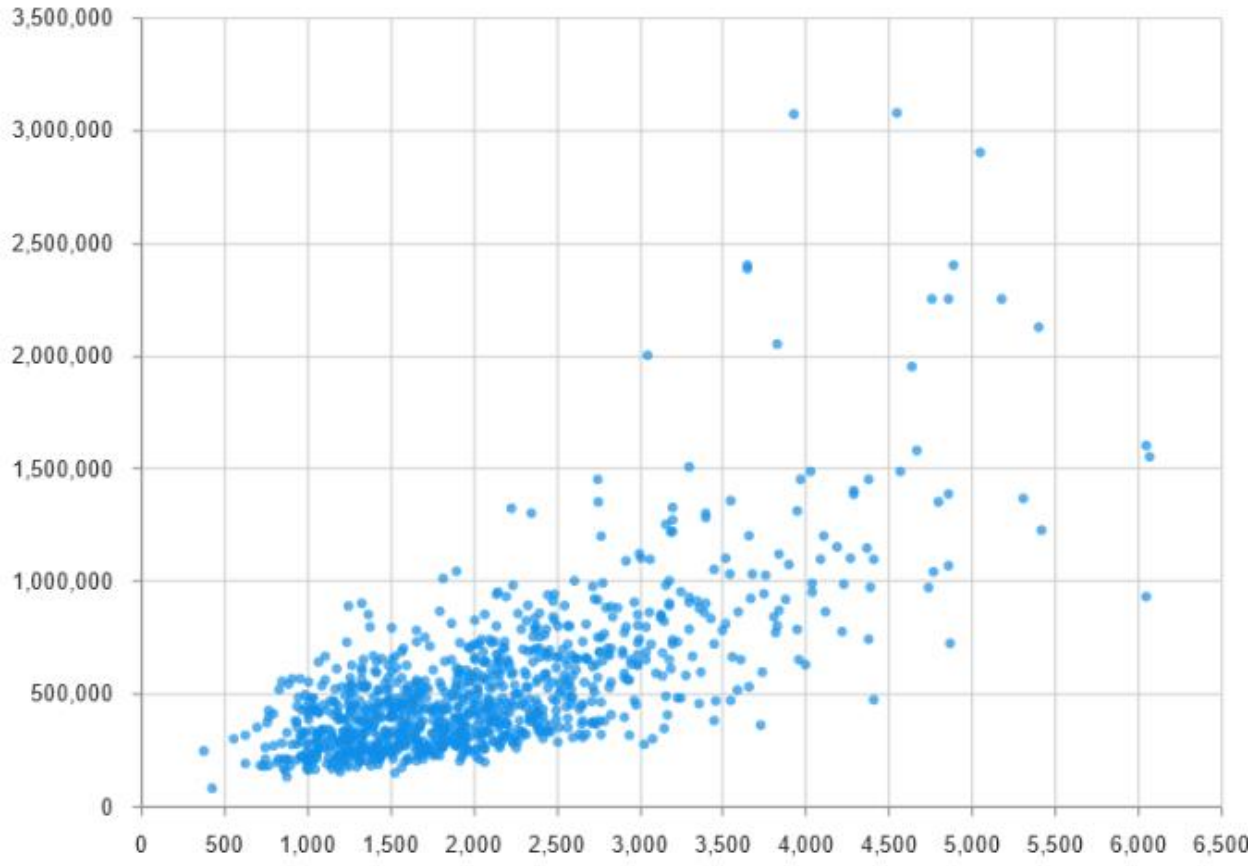


Figure 8: Illustration of the first one thousand samples used in the model representing the house features of price and space

And then for stage 2 we upgraded our features to include (bedrooms, bathrooms, living room area in square feet (sqft), lot area in sqft, floors, zipcode) and used multiple linear regression to build a more accurate model with a smaller RMSE of `180439.07296639`.

The used code in stage 2:

```
my_feartures = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors','zip
code']
model_features = tc.linear_regression.create(train_data,target = 'price',feat
ures=my_feartures,validation_set=None)
print(model_features.evaluate(test_data))
```

For the final stage we added all of the available features that included (condition, waterfront, view, basement area in sqft, the year when the house was built, the year when the house was renovated) in addition to the previous features, and what we

got was a more advanced model with even smaller RMSE of 155269.6579273.

The used code in stage 3:

```
advanced_features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors'
,'zipcode','condition','grade','waterfront','view','sqft_above','sqft_basemen
t','yr_built','yr_renovated','lat','long','sqft_living15','sqft_lot15']
model_advanced = tc.linear_regression.create(train_data,target = 'price',feat
ures = advanced_features,validation_set=None)
print(model_advanced.evaluate(test_data))
```

Conclusion

In this research, we went through the main concepts of machine learning (ML) and specifically supervised ML (regression) and the whole preparation process of Data splitting and its main algorithms, in addition to Data regularization, and we introduced some of many regression methods, but we focused on the ones that deal with real continuous values and mostly the methods used in the practical field of ML. finally, we ended our research with a case study of a real-life regression application in house price predicting.
We hope that we could clarify the concept of using Regression analysis in building ML models, which is one of the hottest trends nowadays.

References

[1] Machine Learning Algorithms - Popular algorithms for data science and machine learning by Giuseppe Bonaccorso, page 404. Available: https://www.packt.com

[2] Mitchell, T. M., Machine Learning, McGraw-Hill, New York, 1997.

[3] Kononenko, I. and Kukar, M., Machine Learning and Data Mining: Introduction to Principles and Algorithms, Horwood Publishing Limited, 2007.

[4] B. Ripley, Pattern Recognition and Neural Networks. Cambridge, U.K.: Cambridge Univ. Press, 2007. [Online]. Available: https://books.google.hu/books?id=m12UR8QmLqoC

[5] G. James, D. Witten, T. Hastie, and R. Tibshirani, An Introduction to Statistical Learning: With Applications In R. New York, NY, USA: Springer-Verlag, 2013. [Online]. Available: https://www.springer.com/gp/book/9781461471370

[6] Efron B, Tibshirani R. An Introduction to the Bootstrap. BocaRaton: Chapman and Hall/CRC; 1993.

[7] Lohr, S. L., Sampling: Design and Analysis, Duxbury Press, 1 edn., 1999.

[8] Efron B. Bootstrap methods: another look at the jackknife. Ann Stat. 1979;7:1–26.

[9] Shao J. Bootstrap model selection. J Am Stat Assoc. 1996;91:655–65

[10] Kennard RW, Stone LA. Computer-aided design of experiments. Technometrics. 1969;11:137–48

[11] Neal B. Gallagher, Donal O'Sullivan, Selection of Representative Learning and Test Sets Using the Onion Method

[12] http://watson.latech.edu/book/future/futureMoores1.html

[13] Andriy Burkov, The Hundred-Page Machine Learning Book, page 23 -24