Steps in Machine Learning

The following are the steps used to perform a Machine Learning Task:

1. Collecting the data

Be it the raw data from excel, access, text files, images, video etc., this step (gathering past data) forms the foundation of the future learning. The better the variety, density and volume of relevant data, better the learning prospects for the machine becomes.

2. Preparing the data

Bad data always lead to bad insights that leads to problems. Any analytical process thrives on the quality of the data used. One needs to spend time determining the quality of data and then taking steps for fixing issues such as missing data and treatment of outliers.

3. Training the model

This step involves choosing the appropriate algorithm and representation of data in the form of the model. In layman terms model representation is a process to represent our real-life problem statement into a mathematical model for the computer to understand. The cleaned data is split into three parts – Training, Validation and Test - proportionately depending on the scenario. The training part is then given to the model to learn the relationship / function

4. Evaluating the model

Quite often, we don't train just one model but many. So, to compare the performance of the different models, we evaluate all these models on the validation data. As it has not been seen by any of the models, validation data helps us evaluate the real-world performance of models.

5. Improving the Performance

Often, the performance of the model is not satisfactory at first and hence we need to revisit earlier choices we made in deciding data representations and model parameters. We may choose to use different variables (features) or even collect some more data. We might need to change the whole architecture to get the better performance in the worst case.

6. Report the Performance

Once we are satisfied by the performance of the model on the validation set, we evaluate our chosen model on the test set and this provides us with a fair idea of the performance of our model on real-world data that it has not seen before.

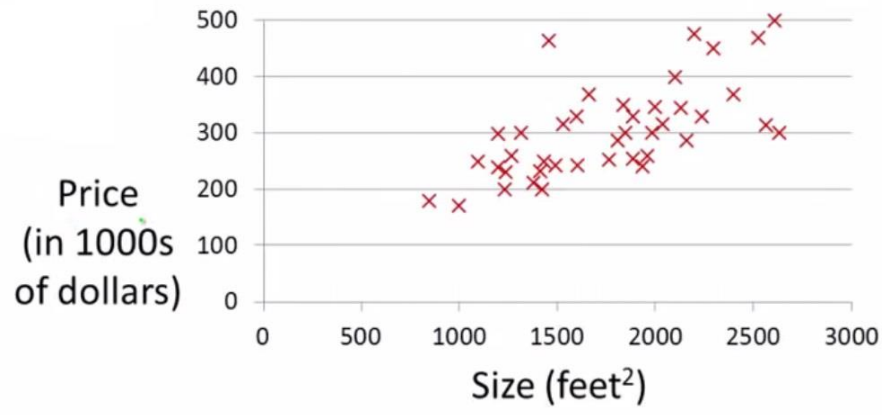
Let's understand this by Supervised learning example of Housing Price Prediction

Example: Suppose, you have **100 houses** in a colony of **different prices**. Let's **assume** for now that price of houses solely depends on **the size**. You must understand what is the relationship of size and price. You can perform classification or regression by understanding the relationship. Let's focus on regression

1. Collecting the data: First you would need to collect the data. In this problem you need to collect the size and prices of all the houses in the colony.

2. Preparing the data: In this example, there is nothing seems to be clean in the data. It all depends on how you collect the data.

3. Training the model: In this step, we need to choose the model and put it into training.
Now, in real world the data will look something like this.



It may be observed that it is impossible to fit a direct mathematical relationship. Thus, there is no direct model on which we can train this data. First, you need to split the data into training, validation and test sets. A rule of thumb is to use 80% of the data for training, 10% for validation and 10% for testing. You can vary this according to the problem. There is no compulsion.

Thus, we need to some choose some model on which we can train this data. Let's say we choose to fit a linear model (i.e. A linear relationship between X and Y) on it. A line will fit overall trend of the data appropriately.
Note: *Graphically, in two dimensions, this results in a line of best fit while in three dimensions, we would refer to it as a plane of best fit, and so on in higher dimensions as a hyperplane of best fit.*

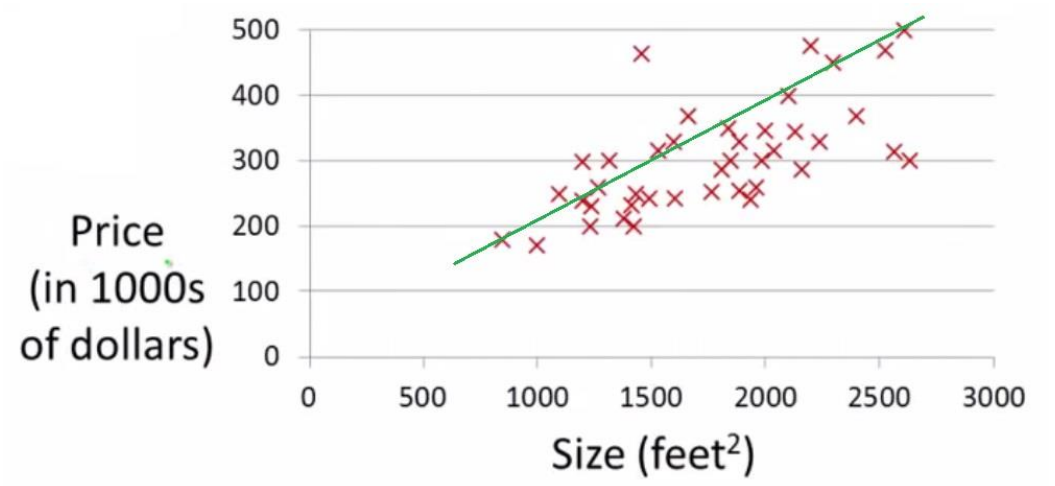
Thus, we can model this problem as

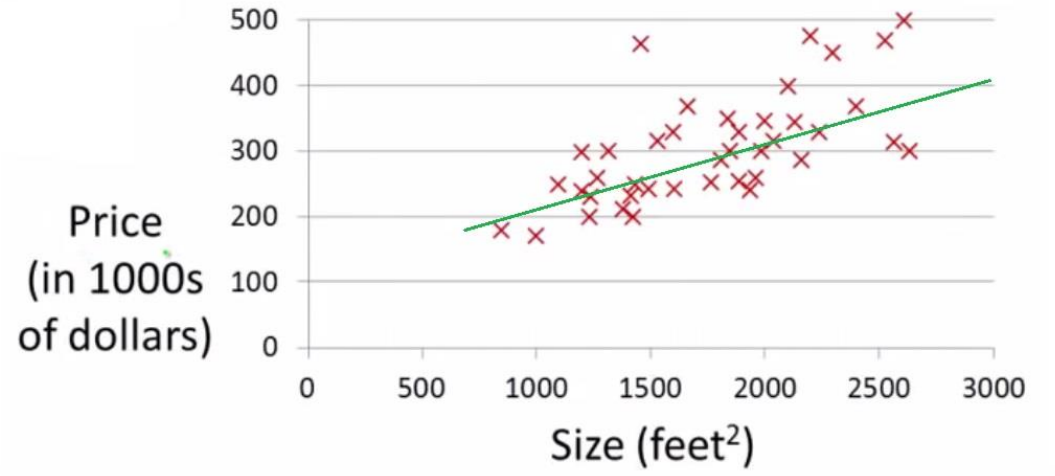
$$y = \theta_0 + \theta_1 * x$$

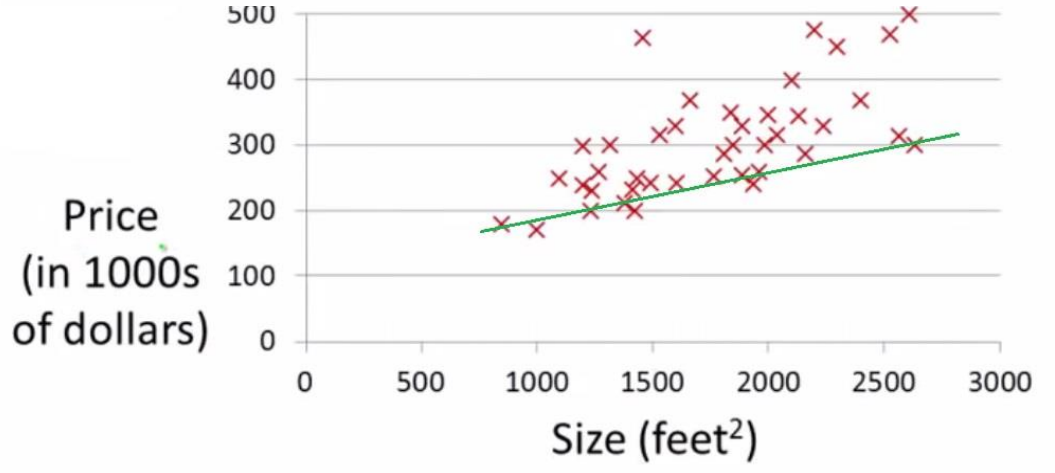
where, y is the price (in 1000s$) and x is size (in units of feet^2) and θ_0 and θ_1 are arbitrary constants.

Our model will be a function that predicts y for a given x and our goal is to learn the model parameters (θ0 & θ1 in our case.) But what are the best model parameters? Because of the noise in the data, there can be many solutions possible for this problem based on different values of θ_0 & θ_1. Let's look at a few of them.

.


Price
(in 1000s
of dollars)
500
400
300
200
100
0
0
500
1000
1500
2000
2500
3000
Size (feet²)


Price
(in 1000s
of dollars)
500
400
300
200
100
0
0
500
1000
1500
2000
2500
3000
Size (feet²)


Price
(in 1000s
of dollars)
500
400
300
200
100
0
0
500
1000
1500
2000
2500
3000
Size (feet²)

Which one of these should be picked? How to learn which one will be the best model parameters (θ_0 & θ_1)?
The best model parameters are those which gives most accurate results. The line which covers most of the trend in data will be most accurate. Alternatively, the best model parameters are those which gives least error. The line which gives least inaccurate results will be best line. We thus choose the line which minimizes error in model's predictions.

The task of finding the model parameters reduces to finding those parameters that minimize error, i.e. make model as accurate as possible. But how do we measure error?
To measure error, we define a error function commonly known as **cost function or loss function** which measures how inaccurate the model's predictions are.

Mathematically, we look at the difference between each real output ($\hat{y}$) and our model's prediction (y). Square these differences to avoid negative numbers and penalize larger differences more than smaller ones, and then add them up and take the average. This is a measure of how well our data fits the line and is commonly known as the mean squared error.

$$Cost = \sum_{1}^{n} \frac{(y - \hat{y})^2}{2 * n}$$

Where "n" is the total training samples. If we substitute the value of y from above equation then we get,

$$Cost = \sum_{1}^{n} \frac{((\theta_0 + \theta_1 x) - \hat{y})^2}{2 * n}$$

For a simple problem like this, we can compute a closed-form solution using calculus to find the optimal parameters that minimize our loss function.

But as a cost function grows in complexity, finding a closed form solution with calculus is no longer feasible and therefore an iterative approach called **gradient descent** is used to minimize more complicated loss functions.

You need to follow the video lectures to understand this thing deeper and understand what exactly the gradient descent is and how we solve this problem.

4. Evaluating the model: Now you evaluate your model on validation set on which your error happens to be minimum.

5. Improving the performance: After training the model, you might feel that you are not getting the best performance. In that case, you might need to change the model. You can change your linear model to some non-linear model like parabolic ($y = \theta_0 + \theta_1 * x + \theta_2 * x^2$) or cubic ($y = \theta_0 + \theta_1 * x + \theta_2 * x^2 + \theta_3 * x^3$) or to some logarithmic ($y = \theta_0 log(kx)$) etc.
But remember nonlinear model are hard to train, they consume much more time, energy and they might not be worth the trade-off between time and performance.

6. Report the Performance: After getting satisfactory performance on validation set, we evaluate our chosen model on the test set and this provides us with a good idea of the performance of our model on real-world data that it has not seen before.

The example we have discussed is a Supervised Learning Problem of Regression. This is popularly known as **“linear regression”.** Since there is only one dependent variable, this has special name called **“Linear Regression in one variable”.**

When you get more than one variable, like along with size, prices also start to depend on the number of rooms, unfurnished/Furnished houses, etc. Then this problem is more challenging than one variable and hence there is also other class of problems called “**Linear Regression in Multiple variables**”

Now the problem might change to Classification in Supervised Learning or even to an Unsupervised learning model or even to the Neural Networks but approach to solve the problem remains same. Only the model changes and thus, consequently the method to solve the model.