Machine Learning for Networking

Seminar 1

Marc Carrascosa Zamacois: marc.carrascosa@upf.edu

Seminar structure

General structure for all seminars:

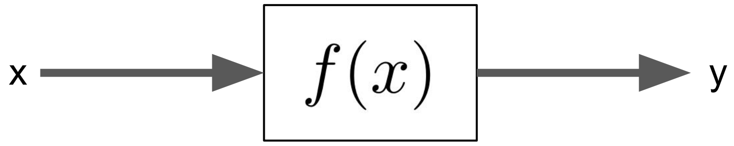
- 30-45 minutes of theory
- 45-60 minutes of work

Seminars count for 20% of the grade, which will be evaluated from a final report/project at the end of the course

While in theory sessions you deal mostly with Reinforcement Learning, in the seminars we will focus on Supervised Learning.

Machine Learning

- Machine learning deals in algorithms or applications that use data to learn by themselves
- Instead of programming a solution for every possible situation, we create a program that understands the context of the situation, finding the solution by itself
  - Example: If we want a program to be able to distinguish between pictures of dogs and cats, we can train it to understand the main features of cats/dogs (ears, tail, whiskers, fur, etc) based on a few examples. Without Machine Learning, the computer would only be able to compare if the pictures are identical to the previous ones it has “seen”.
- X is our data (context) and y is our solution. Machine learning finds a function that connects them

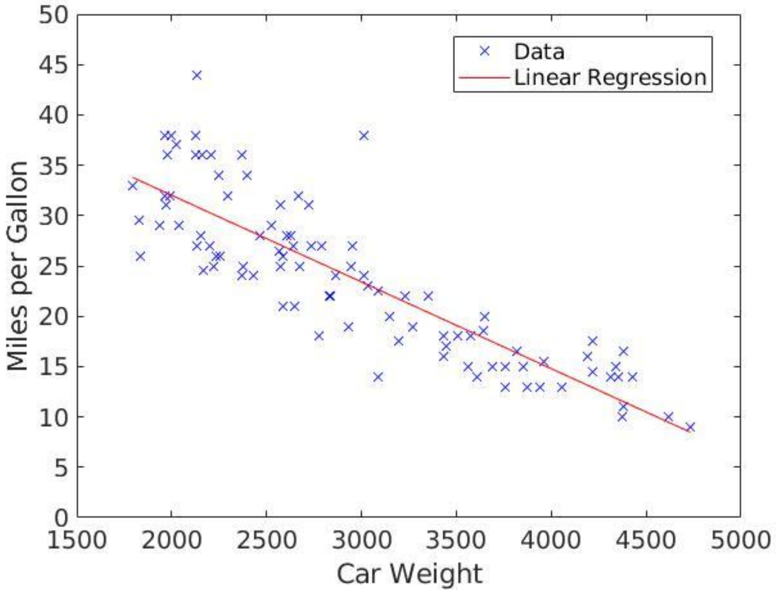


Machine Learning

Some use cases[1]

- Speech recognition like Siri or Cortana
- Image recognition is used to find evidence of lung cancer in CT scans
- Volvo predicts when your car needs repairs
- Google finds phishing attempts automatically
- Netflix predicts what shows you like based on what you watched previously

Example: Miles per gallon (fuel efficiency) of a car based on its weight



1. https://www.forbes.com/sites/bernardmarr/2018/04/30/27-incredible-examples-of-ai-and-machine-learning-in-practice/

How does it work

First, we need data!

Typically, a dataset is presented in the following manner:

Result/Output

Features

Samples

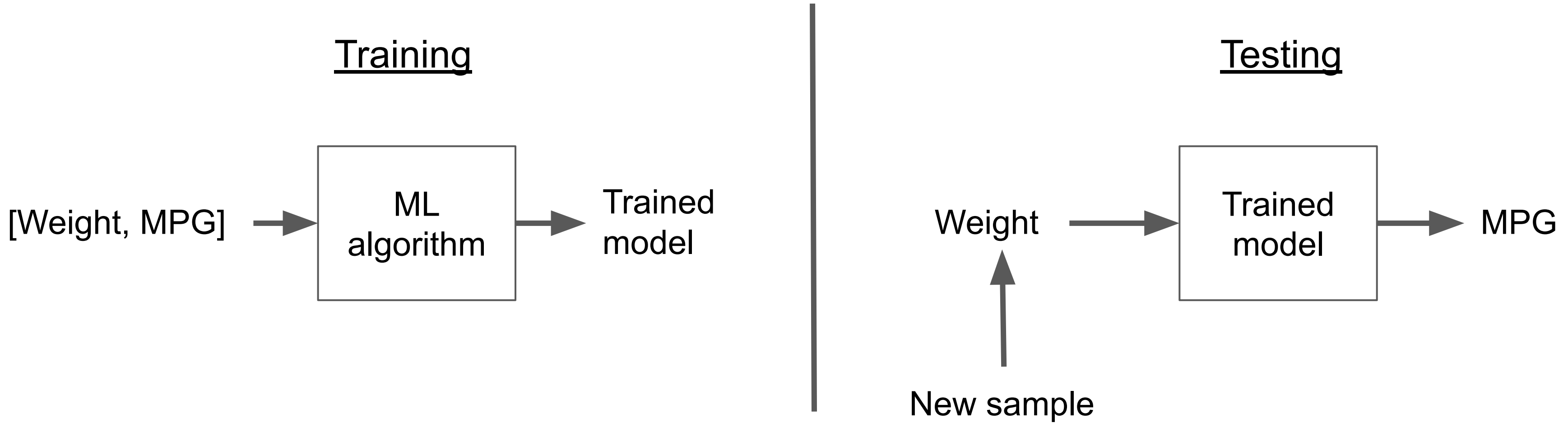
AGE	SEX	BMI	BP	S1	S2	S3	S4	S5	S6	Y
59	2	32.1	101	157	93.2	38	4	4.8598	87	151
48	1	21.6	87	183	103.2	70	3	3.8918	69	75
72	2	30.5	93	156	93.6	41	4	4.6728	85	141
24	1	25.3	84	198	131.4	40	5	4.8903	89	206
50	1	23	101	192	125.4	52	4	4.2905	80	135

https://azure.microsoft.com/es-es/services/open-datasets/catalog/sample-diabetes/

Using the dataset

To train a Machine Learning algorithm, we feed it samples in pairs (x,y)

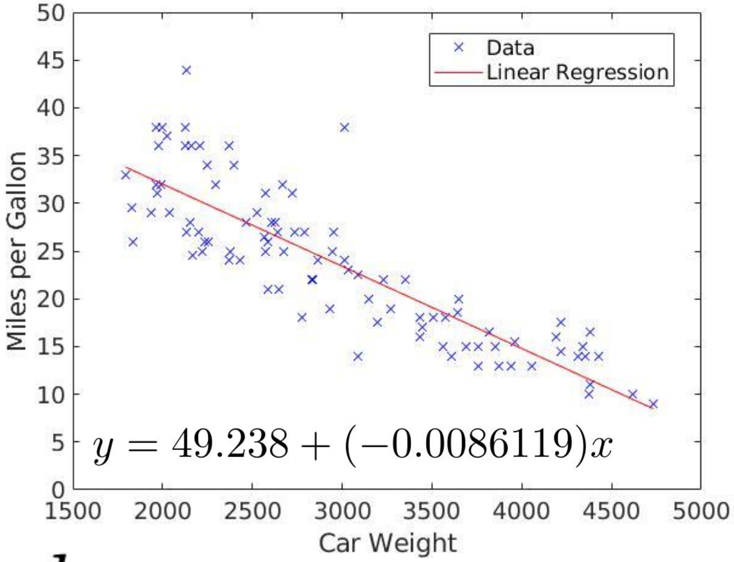
- X can be a matrix of as many columns (features) as we want
- Y is the result, what we want the algorithm to be able to predict in the future, and It needs to have as many samples as X



Linear Regression

As its name suggests, it finds a way to represent data that has a linear relationship in the shape of:

$$y = b_0 + b_1 x$$



And now if we want to check for a new car that weighs 3300 pounds, we get 20.81873 MPG

What if we have more than one feature?

$$y = b_0 + b_1 x_1 + b_2 x_2 + \ldots + b_n x_n$$

Linear regression

To calculate the regression coefficients then, we first present the system as

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & \cdots & x_{n,1} \\ 1 & x_{1,2} & x_{2,2} & \cdots & x_{n,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1,m} & x_{2,m} & \cdots & x_{n,m} \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}$$

With

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \qquad X = \begin{bmatrix} 1 & x_{1,1} & x_{2,1} & \cdots & x_{n,1} \\ 1 & x_{1,2} & x_{2,2} & \cdots & x_{n,2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1,m} & x_{2,m} & \cdots & x_{n,m} \end{bmatrix} \qquad \vec{b} = \begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_n \end{bmatrix}$$

Linear Regression

Linear regressions work by minimizing the squared error

$$Err(\hat{y}, y) = \hat{y}^2 - y^2$$

Where y is the real value in the data, and $\hat{y}$ is the predicted value

Finally, we combine our formulation with the least squares error to obtain

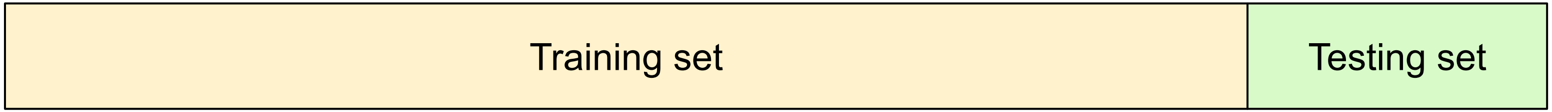
$$\vec{b} = (X^T X)^{-1} X^T \vec{y}$$

Training and testing a model

Before training the model, the dataset needs to be split into training and testing sets

- Training set is used to train the model
- Testing set is used to check the accuracy of the model’s predictions

This serves to ensure that the model is not excessively tuned to the training data, and that it will accurately represent the target function we aim to model



Testing and calculating the error

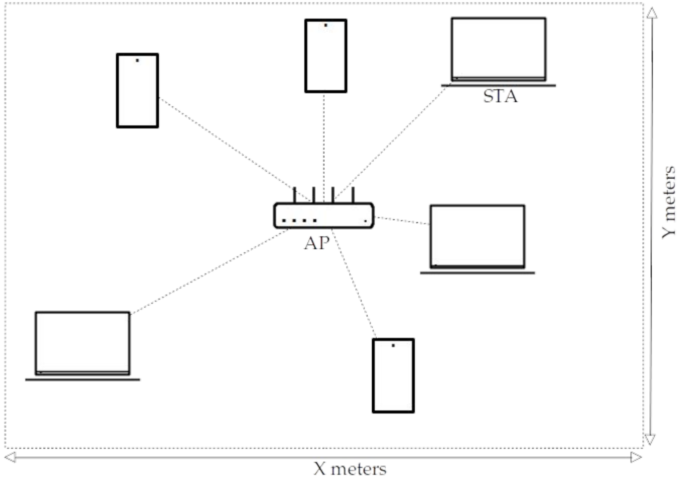
For Linear Regression, it is common to use the Root Mean Square Error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{n=1}^{N}(\hat{y}_n - y_n)^2}{N}}$$

Where we feed the model with the testing set, and then compare the testing y to the predicted $\hat{y}$

Our dataset

- Maps the performance of a single WiFi network
- Rows are samples
- Columns are features
- We will study how to predict the performance of a network based on AP information



Dataset - inputs

1. **Number of STAs:** 1-40
2. **Load:** 0.5 - 82 Mbps
3. **Size(x):** 1-40 metres
4. **Size(y):** 1-40 metres
5. **Area:** x·y
6. **Contention window:** {3, 15, 31, 63, 127, 255, 511, 1023} slots
7. **Channel width:** {20, 40, 80, 160} MHz
8. **Packet size:** {4000, 6000, 8000, 10000, 12000} bits
9. **Max RSSI:** ≥ -82 dBm
10. **Avg. RSSI:** ≥ -82 dBm
11. **Min. RSSI:** ≥ -82 dBm

Dataset - outputs

1. **Avg. Probability of failure:** 0-1
2. **Throughput:** Aggregated over the whole network, in bps
3. **Average delay:** Average transmission time of a packet, in seconds
4. **Total airtime:** Sum of transmission time necessary for all transmissions
5. **Proportional airtime:** Available transmission time used by the network

For this seminar we will look mainly at the throughput of the network

Dataset analysis

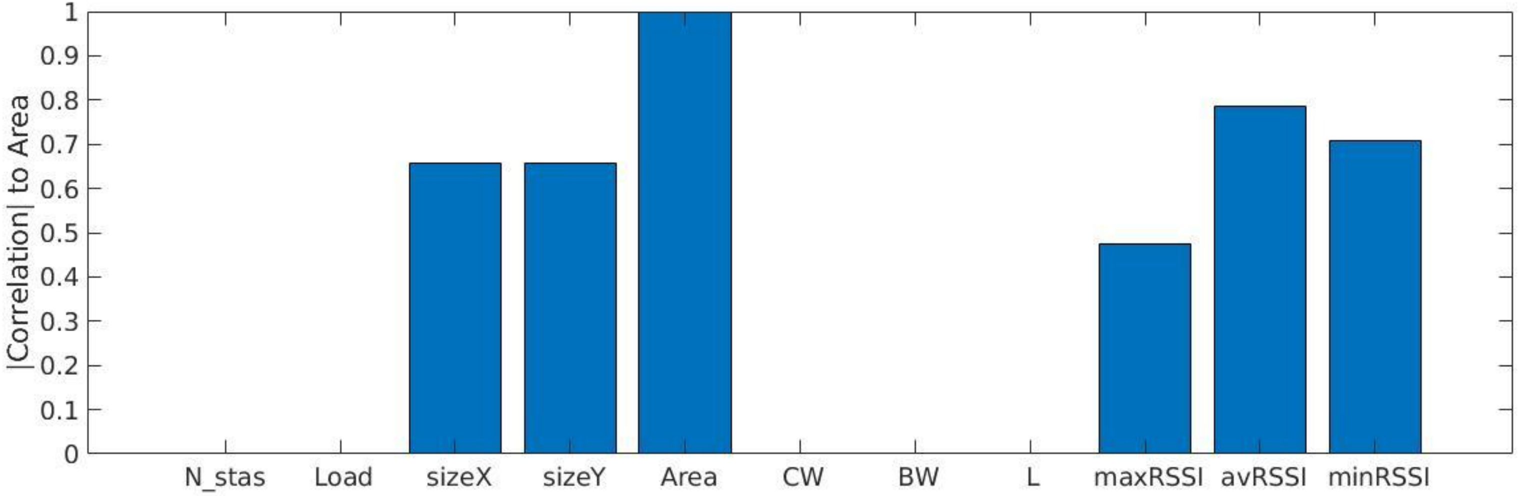
A dataset can have many features, and their relevance to predicting the output is not always equal. What do you think is more relevant when predicting throughput? Is the load of the network important for such a prediction? What about the area of the network?

To get an idea about the importance of each feature, we can study the correlation of the features (you can use the function *corrcoef*)

Feature analysis

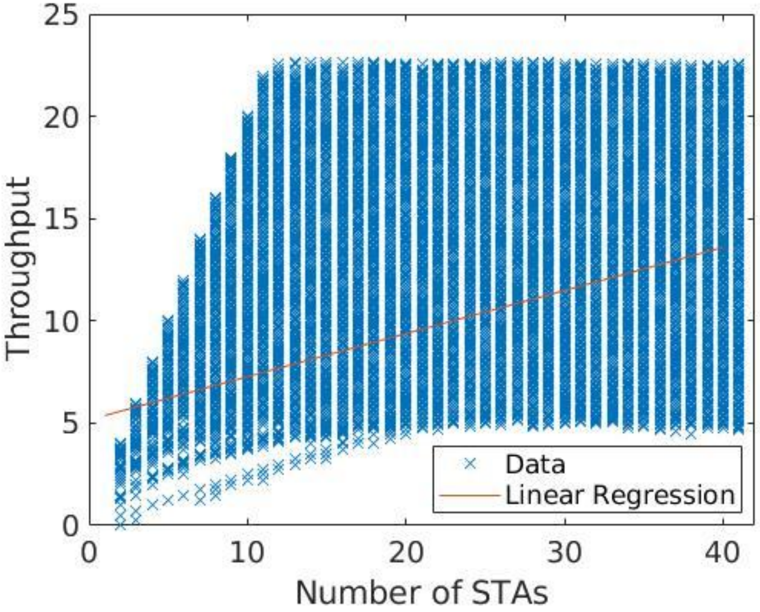
Example: the correlation between all the features and the area occupied by the network.

- The horizontal and vertical size of the area obviously hold significance when calculating the area. But the size of packets does not.
- Does it make sense that the average RSSI is correlated with the area size?



Linear Regression

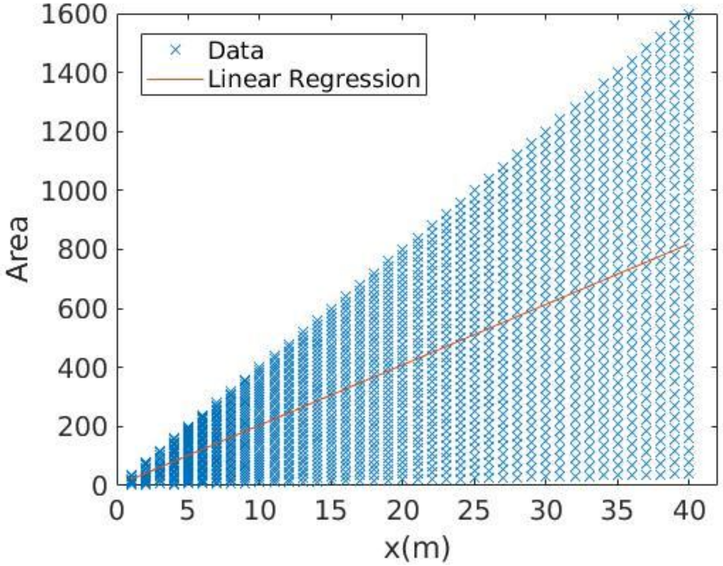
Do we need more than one feature? The following example shows a regression of our dataset, in which we only use the number of STAs as our input to predict the throughput. Is it accurate enough?



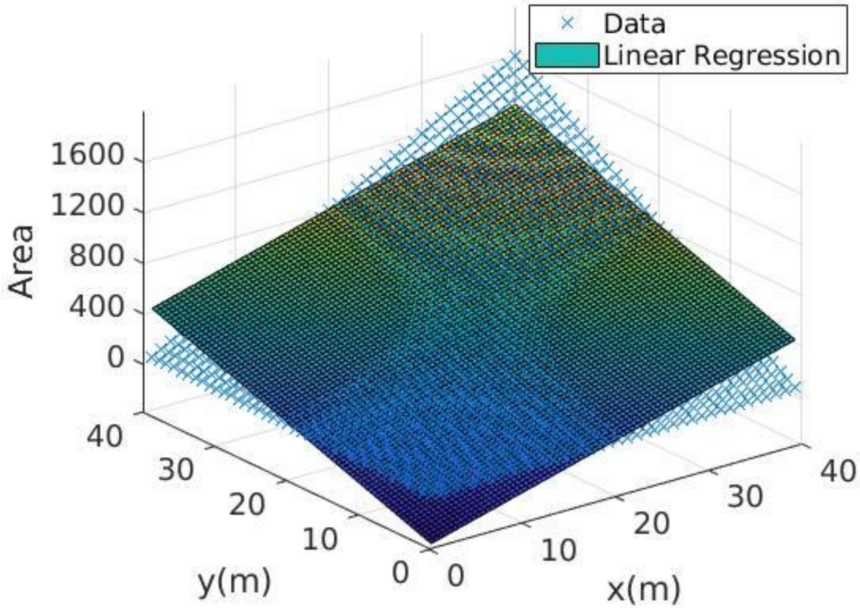
In general, the more features we have the better, as each one adds unique information. If a feature is completely irrelevant, once we perform the regression its coefficient will be 0.

Linear Regression

Example: Linear Regression of the area size, based on one or two features

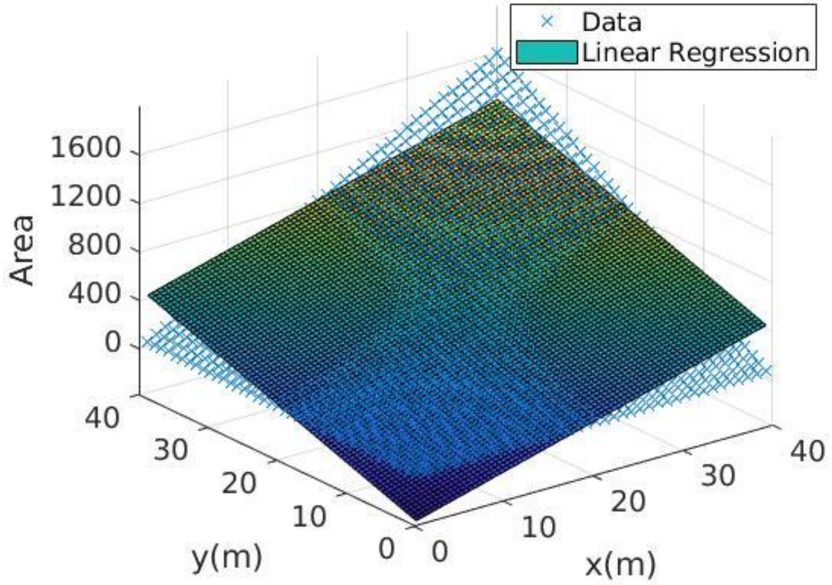


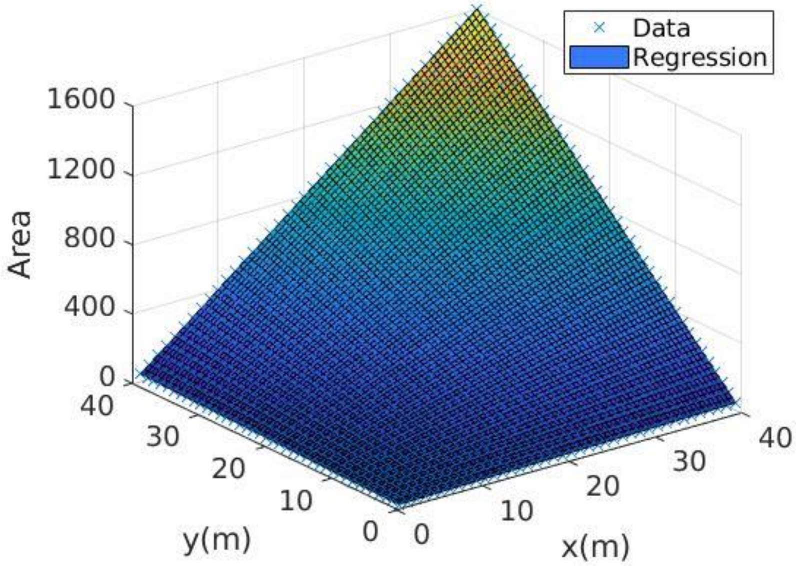
RMSE = 271.601



RMSE = 133.200

Data (and regression) is not always linear!





$$y = b_0 + b_1 x + b_2 y$$

$$y = b_0 + b_1 x + b_2 y + b_3 xy + b_4 x^2 + b_5 y^2$$

RMSE = 133.200

RMSE = 0

Tips

Useful MATLAB functions for this seminar:

- Corrcoef (for correlation coefficients)[1]
    - Coefficients = **corrcoef**(dataset_matrix)
- Fitlm (for model fitting)[2]
    - Mdl = **fitlm**(training_input, training_output)
- Predict (get predictions based on the fitlm model)[3]
    - Predicted_values = **predict**(Mdl, testing_input)

1. https://mathworks.com/help/matlab/ref/corrcoef.html
2. https://mathworks.com/help/stats/fitlm.html
3. https://mathworks.com/help/stats/linearmodel.predict.html