Coding w/ Vihaan

A coding and computers channel.

Artificial Intelligence & Machine Learning

Applications:

- Personalised Recommendations
- Fraud Detection
- Financial Trading
- Healthcare Diagnostics
- Supply Chain Optimisation
- Image / Speech Recognition
- Autonomous Vehicles

Concepts:

- Introduction to AI/ML
- k-Nearest-Neighbour Classification
- Linear Regression
- Loss Function
- Overfitting & Regularisation

Python Libraries:

`scikit-learn`

Introduction to AI/ML

Introduction to AI/ML

- Artificial intelligence (AI) — intelligence exhibited by machines and computer systems. It is a field of research in CS (computer science) that develops methods and software enabling machines to use learning and intelligence to take actions that maximise their chances of achieving defined goals.

- Machine learning (ML) — a field of study in within AI about the development of statistical algorithms that can learn from and generalise data, and therefore perform tasks without explicit instructions.

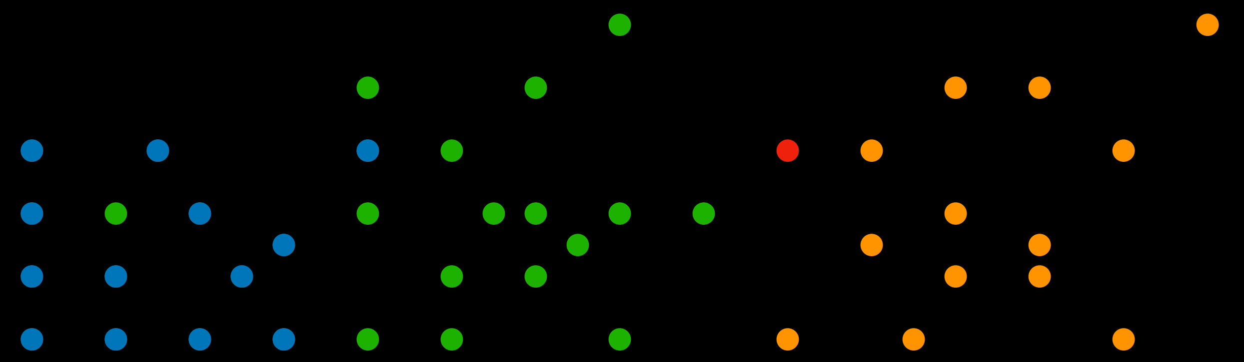
k-Nearest-Neighbour Classification

k-Nearest-Neighbour Classification

- k-nearest-neighbour classification (k-NN) is a non-parametric supervised learning method used for classification and regression.
- The input consists of the k closest training examples in a data set.
- The output is a class membership.
- An object is classified by a majority of its neighbours. The object will be assigned to the class most common among its k nearest neighbours, where k is a positive, usually small, integer.
- For example, if k = 1, then the object is assigned to the class of that single nearest neighbour.

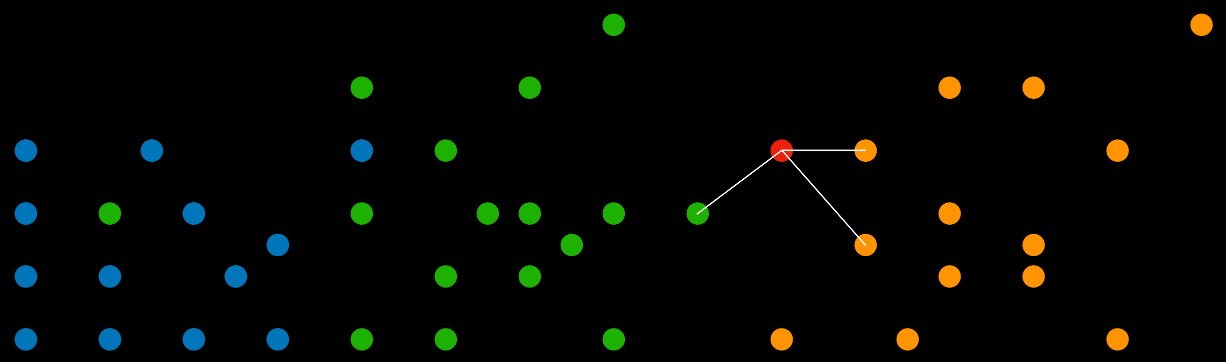
k-NN Classification Example

- Take the below graph. We need to categorise the red point.
- We will run k-NN with k = 3.



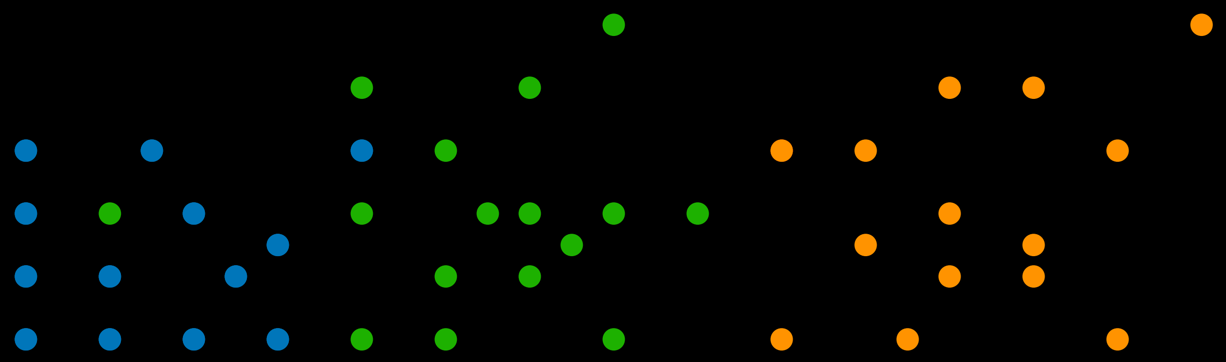
k-NN Classification Example

- Since k = 3, we need to identify the 3 nearest neighbours. They are yellow, yellow, and green.



k-NN Classification Example

- The majority of the neighbours are yellow, so we categorise the new point as yellow.



k-NN Classification Code

- See the code at https://github.com/VihaanAnand/Coding-Vihaan/blob/main/artificial-intelligence-machine-learning/knn.py

Linear Regression

Linear Regression

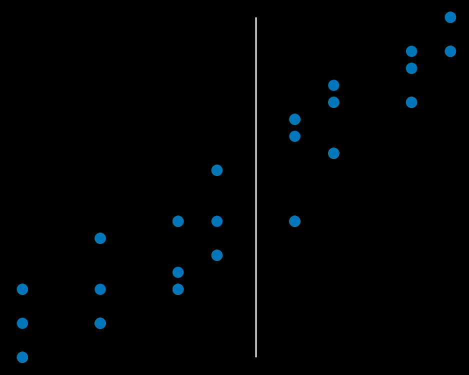
- Linear regression is an algorithm that learns from the labelled datasets and maps the data points to the most optimised linear functions that can be used for prediction on new datasets.

- Linear regression can be represented in the form $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$, where $x_1, x_2, \ldots, x_n$ are the independent variables, and y is the dependent variable.

- Linear regression models are usually fitted using the least squares approach, but they may also be fitted in other ways, such as by minimising with ridge regression (L_2-norm penalty), lasso regression (L_1-norm penalty), or Mean Squared Error (MSE).

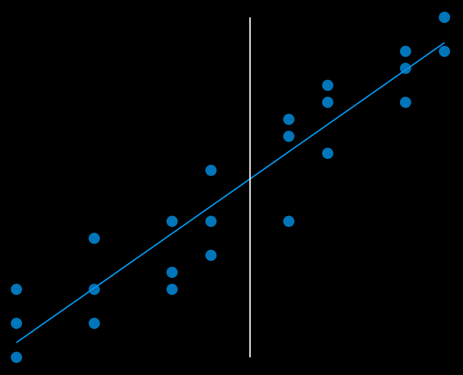
Linear Regression Example

- Assume we are given the data set shown below, and we need to predict the value of the function at a given x-value.



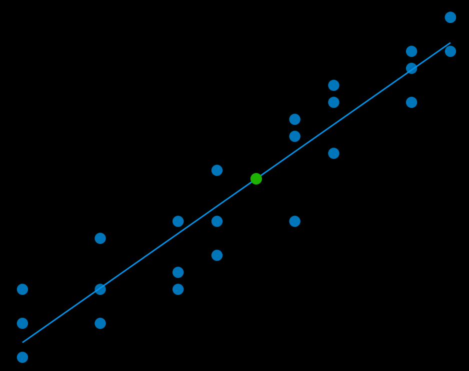
Linear Regression Example

- We would use linear regression to identify a linear pattern in the data, as shown in the graph below.



Linear Regression Example

- Using the linear pattern, we would see what the value of y is at the given value of x. This point can be plotted on the graph below.



Linear Regression Code

- See the code at https://github.com/VihaanAnand/Coding-Vihaan/blob/main/artificial-intelligence-machine-learning/linear.py

Loss Function

Loss Function

- A loss function (or cost function or error function) is a function that maps values of variables onto a real number representing some "cost" associated.

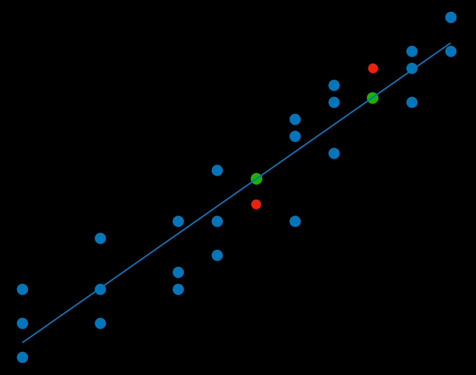
- In linear regression, we don’t measure accuracy. Rather, we use various loss functions, and try to minimise the value of the loss function to show a more accurate model.

Loss Function

- There are many different kinds of loss functions. Here are a few popular ones:
- 0-1 Loss Function. In this loss function, either the answer is wrong (1), or correct (0). Add all the values to get the total loss. The idea behind this function is that values should be exact.
- L_1 Loss Function (Lasso). In this loss function, we take the difference between each predicted answer and actual answer, and add them all up. In this function, it is important to maintain accuracy, no matter how much the error is.
- L_2 Loss Function (Ridge). In this loss function, we determine the difference between each predicted answer and actual answer. After that, we square each of those differences, and add them all up. This function penalises items with higher difference more than those with a lower difference.

Loss Function Example

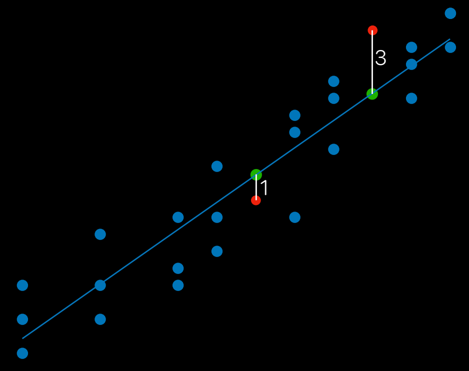
- Let's go back to our Linear Regression example from earlier. I have drawn a few more predicted points, and highlighted the actual points in red.



Loss Function Example

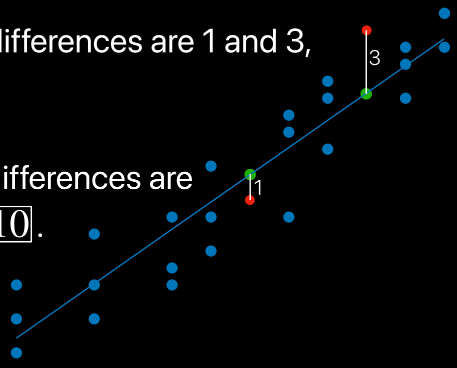
- We will determine the difference between the actual value and predicted value using lines.

- The differences are 1 and 3.



Loss Function Example

- With different loss functions, the loss is different.
- 0-1 Loss - Both predictions are wrong, so $1 + 1 = \boxed{2}$.
- L_1 (Lasso) Loss - The differences are 1 and 3, so $1 + 3 = \boxed{4}$.
- L_2 (Ridge) Loss - The differences are 1 and 3, so $1^2 + 3^2 = \boxed{10}$.



Loss Function Code

- See the code at https://github.com/VihaanAnand/Coding-Vihaan/blob/main/artificial-intelligence-machine-learning/loss.py

Overfitting & Regularisation

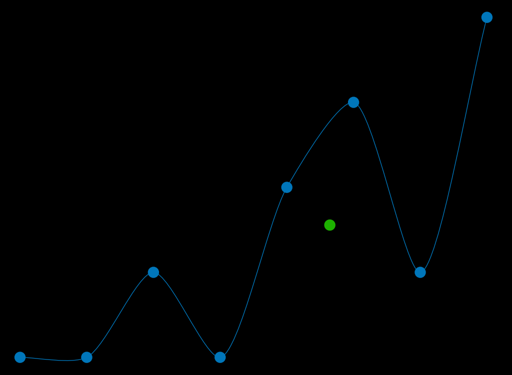
Overfitting & Regularisation

- Overfitting is an analysis that corresponds too closely or exactly to a particular dataset, and may therefore fail to fit to additional data accurately or predict future observations reliably.

- Regularisation is a process that converts a model to a simpler one by penalising complexity of the model. It is often used to prevent overfitting.

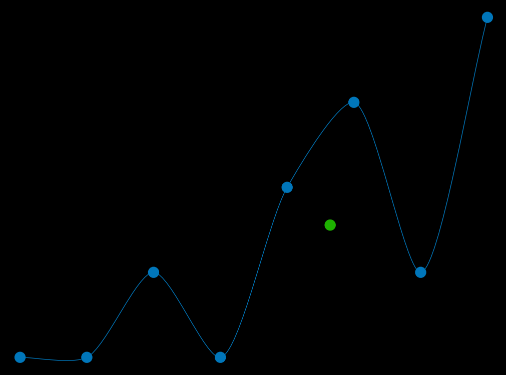
Overfitting & Regularisation Example

- The below graph is an example of overfitting. While it will have zero loss on the points given, it will fail when trying to predict another point, like the one graphed in green. The green point is not far off from the linear regression line.



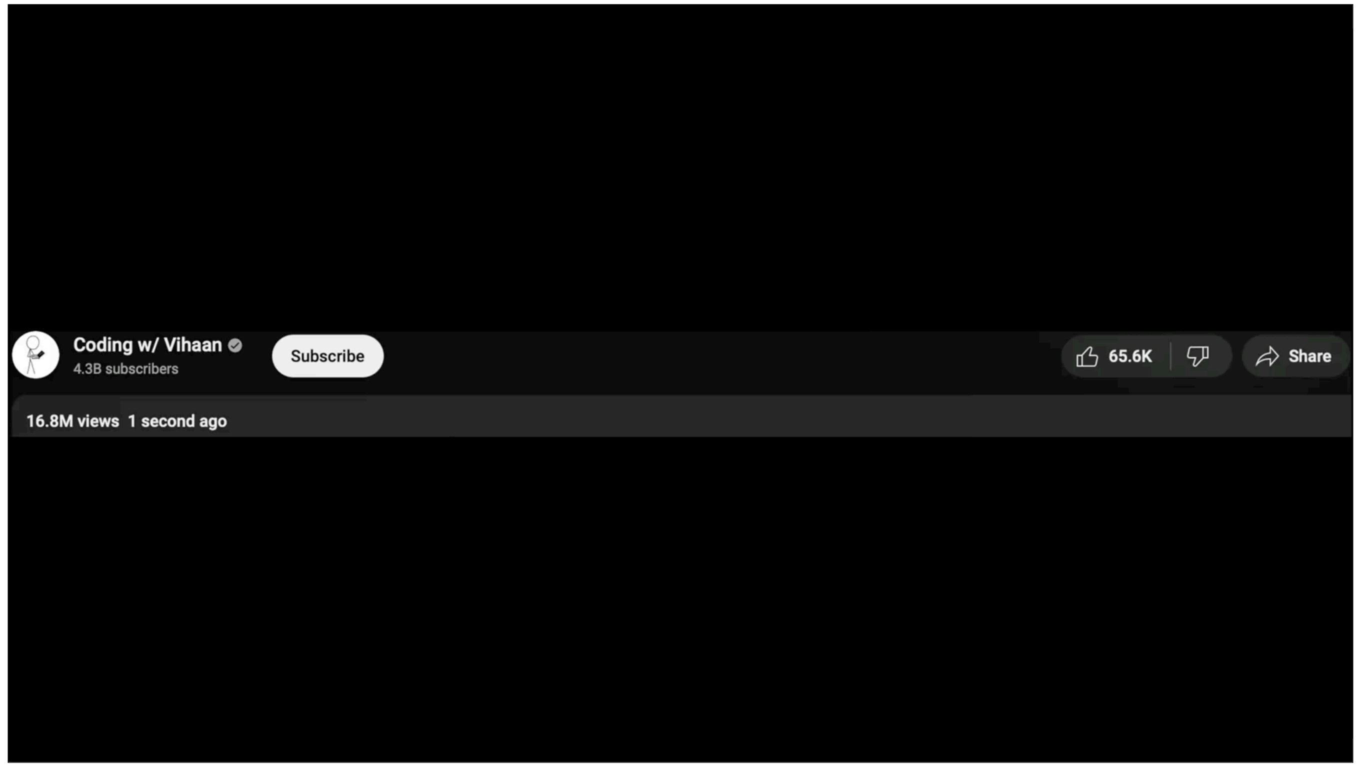
Overfitting & Regularisation Example

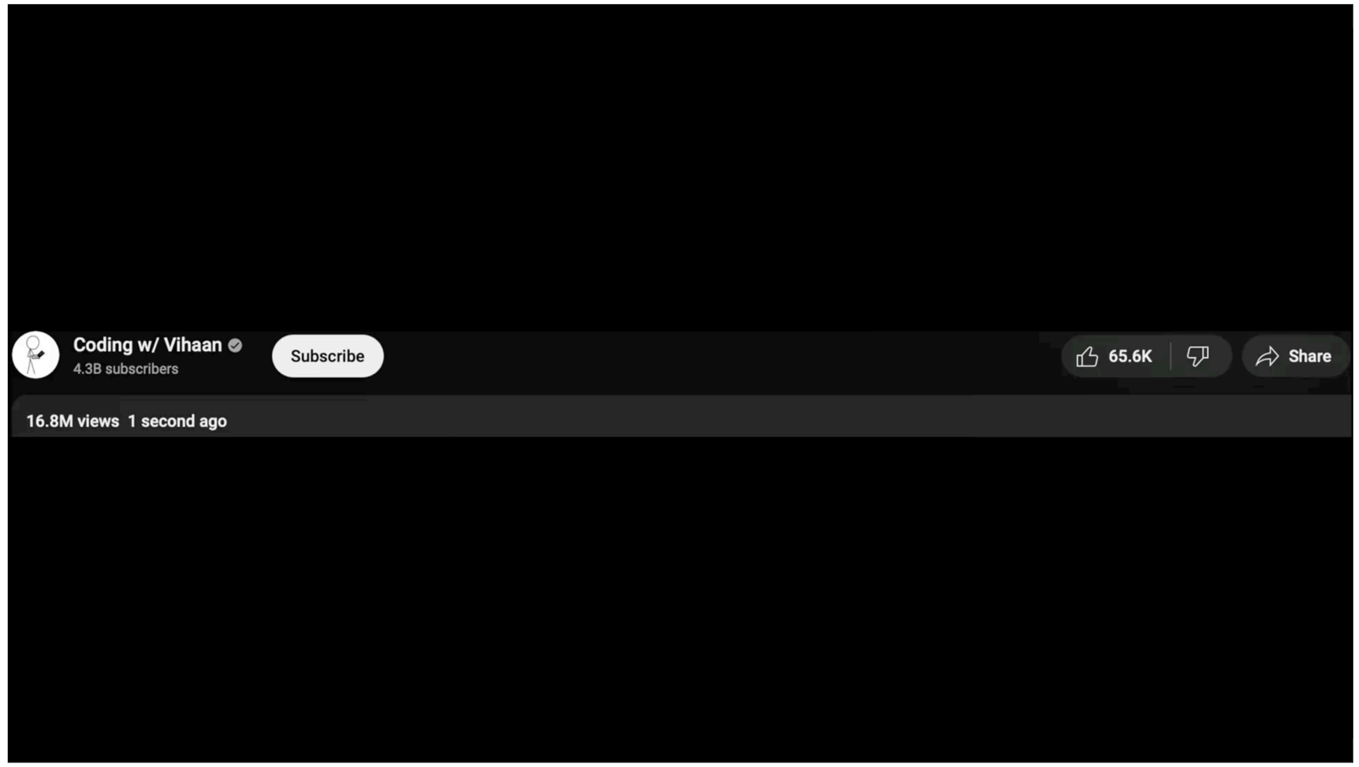
- Regularisation is a calculation that penalises complexity of the model. For instance, this model would have a significantly higher complexity than a straight line.



Overfitting & Regularisation Code

- See the code at https://github.com/VihaanAnand/Coding-Vihaan/blob/main/artificial-intelligence-machine-learning/ofreg.py


Coding w/ Vihaan
4.3B subscribers
Subscribe
65.6K
Share
16.8M views 1 second ago


Coding w/ Vihaan
4.3B subscribers
Subscribe
65.6K
Share
16.8M views 1 second ago