- ***Machine learning algorithms which come with tensor flow***
  - -> these are used separately and combined together
  - ***Models***
    - -> linear regression
    - -> classification
    - -> clustering
    - -> hidden Markov models
    - -> there are more different types and ways of doing them, but those are the main ones
      - there is also more complicated syntax -> which you don't have to remember
      - -> you just have to know how to use it
      - -> tensor flow is a large Python library
  - ***Linear regression***
    - ***Concept***
      - -> drawing lines of best fit and using them to predict things
        - given the x value, you want the y value
        - assuming that x and y are linearly correlated
      - -> the dataset is the points which the line of best fit was drawn through
      - -> either the elements lie above or below the line of best fit
      - -> you can have n dimensional lines of best fit -> e.g 9 dimensional values
      - -> you can do this for a 3D dataset
    - ***Maths***
      - -> it's linear regression <- y = mx + c
      - -> c is the intercept of the line -> the value of y when x is 0
      - -> m is the gradient, dy/dx <- you can draw a triangle on the line
        - -> or calculate the y and x distance between two sets of points
      - -> there are equations you can use for c and m given a dataset
      - -> minimising the total square of the distance from the points above the line to the points below the line
      - -> the points above the line and then the points below the line
      - -> he's done an example of one of these lines
        - you can take the equation for y = mx + c, substitute in a value of x and then get a value of y
        - -> in other words it's predicting a value of y given a value of x
        - -> it's a predictive model -> which is linear
      - -> this can be done in higher dimensions -> e.g given 8 or 9 input variables
        - -> but in this cast the input variables need to be linearly correlated
        - -> the line can be in 3D space and can be linear
        - -> you have variables which are correlating in 3D space
        - -> once you have the equation of the line of best fit and two values, for a line in 3D space then you can predict the third value
      - -> the points had to be correlated linearly
      - -> you could also predict life expectancy -> dependent on their current age
        - predicting their life expectancy based off of if they had an illness
        - -> the entire thing is probabilities and what you would expect the lines to look like for the models

- ***Code***
  - -> !pip install -q sklearn
  - -> sci kit learn
  - -> then we have imports

```
[ ] %tensorflow_version 2.x  # this line is not required unless you are in a notebook

from __future__ import absolute_import, division, print_function, unicode_literals

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from IPython.display import clear_output
from six.moves import urllib

import tensorflow.compat.v2.feature_column as fc

import tensorflow as tf
```

    - numpy is for optimised arrays in Python
      - -> cross, dot products
      - -> vector operations
      - -> matrix multiplication
    - pandas
      - -> manipulating data / datasets
      - -> slicing datasets
      - -> visualising datasets
    - matplotlib
      - -> for graphing different aspects of datasets
    - -> clear_output is cleaning the output of the datasets
    - -> then importing tensor flow linear regression modelling modules