1. **Data preparation**

**Numpy:** allows us to work with arrays.

**Matplotlib:** allows us to plot charts and graphs

**Pandas:** allows us to plot the dataset and create a matrix of features or preprocess your dataset

**Features/independent variables –** variables containing the properties that you will use to make a prediction which is your **dependent variable vector.**

**iloc (locate indexes) –** take the indexes of the columns and rows we want to extract from the dataset. A trick to take all the rows is by using a **colon** (**:**) which stands for a range. To separate the rows from the columns, use a **comma**

A range in python includes the lower bound but includes the upper bound

dataset.iloc[:, :-1].values : in iloc the order is rows, columns

**ct = ColumnTransformer(transformers=, remainder=”passthrough”) : transformers-** Specifies the kind of transformation we want to do and on which indexes of the columns we want to transform, remainder – specify that we actually want to keep the columns that won’t be applied some transformations. **Passthrough** enables us to **not only** keep the three columns resulting from one hot encoding eg 100, 010, 001

**fit\_transform** method doesn’t return it’s output as a numpy array whereby models as a must need the data to be in the form of a numpy array, so you use

**Feature scaling**: scaling all your variables or features to make sure they all take values in the same scale.

Don’t do feature scaling before splitting the dataset to prevent information leakage to the test set

**Note: Standardization works all the time –** puts values of features between -3 and +3 or -2 and +2, while **normalization works when you have a normal distribution of your values –** puts values between 0 and 1

1. **Simple Linear Regression – one independent variable / feature vector**

With **regression** you predict a continuous value while with classification you predict a category.

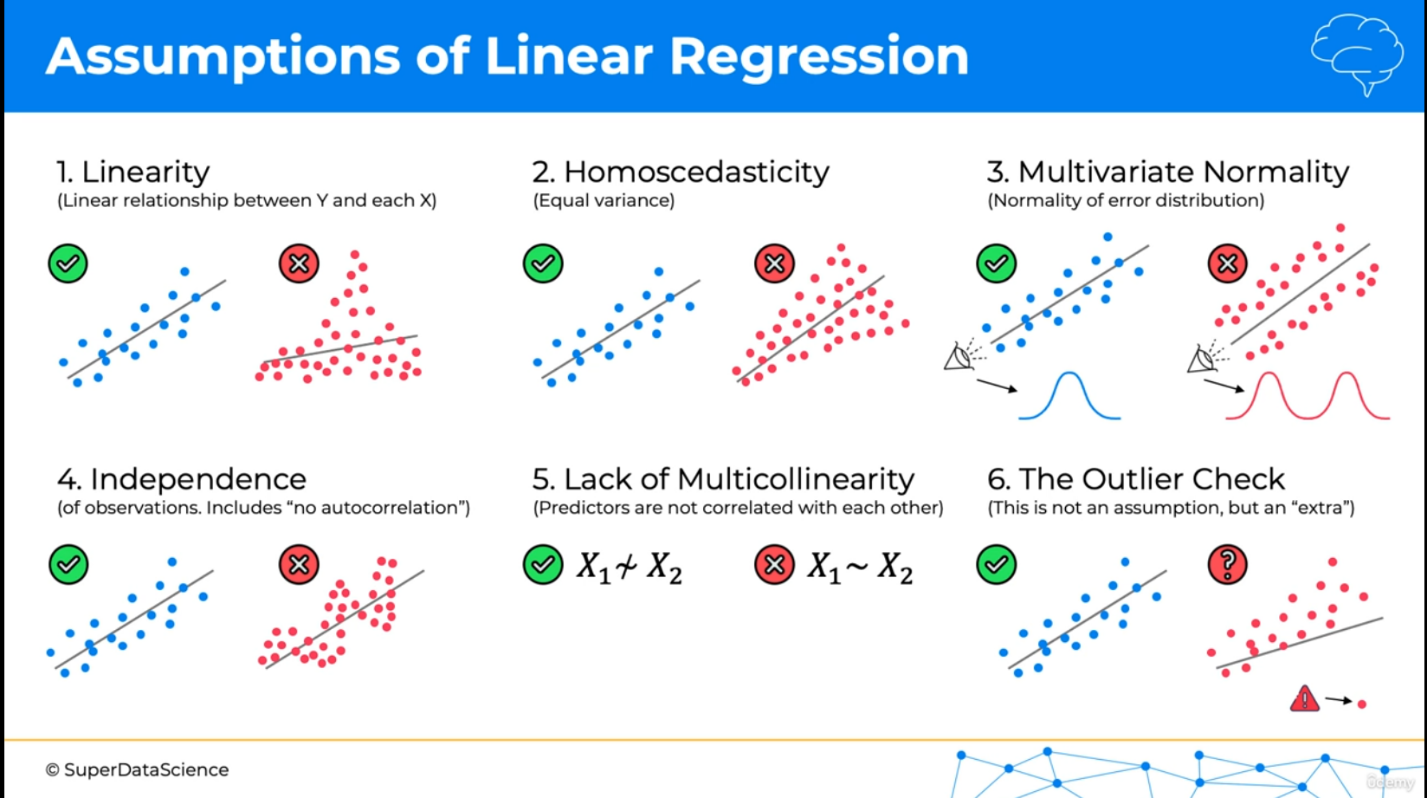
**The best regression line** is the one where the **SUM(y1-y1`)2** is minimized

**Note: (y1-y1`) is the residual that is the difference between the actual point and the predicted point**

1. **Multiple Linear Regression**

**Anscombe’s quartet:** You can’t simply apply linear regression, you must make sure your dataset is fit for using linear regression.

**Assumptions of linear regression**

****

**Independence –** shouldn’t see a pattern in our data eg forex data

**When** you have categorical column data as one of your independent variables, then it’s necessary to use **Dummy variables.**

**Multicollinearity:** situation where one or more independent variables predict one another

**Whenever you’re building a model** only include one dummy variable

**Statistical significance**

**5 methods of building models**

1. All-in
2. Backward Elimination
3. Forward selection
4. Bidirectional Elimination
5. Score comparison

**NOTE:** Stepwise regression refers to 2, 3, and 4 and sometimes just 4.

1. **All-in** – use all your independent variables

**When would you do this**;

* **Prior knowledge**
* **You have to**
* **Preparing for backward Elimination**

1. **Backward Elimination**

**Step 1.** Select a significance level to stay in the model eg SL = 0.05

**Step 2.** Fit the full model with all possible predictors

**Step 3.** Consider the predictor with the highest P-value. If P > SL, go to STEP 4, otherwise go to FIN (finish ie model is ready)

**Step 4.** Remove the predictor

**Step 5.** Fit model without this variable

**After step 5, go back to Step 3**

1. **Forward Selection**

**Step 1.** Select a significance level to enter the model (e.g SL = 0.05)

**Step 2.** Fit all simple regression models y-xn Select the one with the lowest P-value

**Step 3.** Keep this variable and fit all possible models with one extra predictor to the one you already have

**Step 4.** Consider the predictor with the lowest P-value. If P < SL, go to STEP 3, otherwise go to FIN: Keep the previous model

1. **Bidirectional Elimination**

**Step 1:** Select a significance level to enter and to stay in the model

**Step 2:** Perform the next step of Forward selection (new variables must have: P < SLENTER to enter)

**Step 3:** Perform ALL steps of Backward Elimination (old variables must have P < SLSTAY o stay)

**Step 4:** No new variables can enter and no old variables can exit

1. **All Possible Models**

**Step 1:** Select a criterion of goodness of fit (e.g. Akaike criterion)

**Step 2:** Construct All possible regression models: 2N – 1 total combinations

**Step 3:** Select the one with the best criterion

**NOTE:** In multiple linear regression there is absolutely no need to apply feature scaling and also to check for the assumptions of linear regression

1. **Polynomial linear regression**

Handy when trying to determine how epidemics spread over an area

Though the x values are polynomial, we focus on expressing the coefficients as linear values and that’s why it’s called Polynomial **Linear** regression

1. **Linear Support Vector Regression**

Vectors that support the formation of the Epsilon tube or Error tube

1. **Non Linear Support Vector Regression**

**Kernels –** can learn linear relationships ie Linear Kernels or **Non- Linear Relationships ie** Non – linear kernels eg Gaussian Radial Basis Function RBF, Polynomial Kernel, and Gaussian Kernal

1. **CART – Classification and Regression Trees –** you don’t have to use feature scaling on decision tree regression or random forest regression
2. **Essemble Learning** – take multiple algorithms or the same algorithms multiple times to make something more powerful.

**CLASSIFICATION**

1. **Logistic regression:** predicting a categorical dependent value from a number of independent variables. It uses a sigmoid curve to predict data.

**Likelihood = Probability of saying yes \* Probability of saying no**

The logistic regression curve is the one with the **maximum likelihood**

1. **Support Vector Machine:** In SVM the vectors, represent the extreme cases in our dataset.
2. **KNN**
3. **Kernel SVM:** evolves around mapping to a higher dimensional space in order to obtain our non-linear separator. Used when you have categories that are not linearly separable.

**When** you map from a one dimensional space to a 2D space, the separator is a line, when you map from a 2D space to a 3D space, the separator is a hyperplane.

**Problem:** Mapping to a higher dimensional space can be very compute intensive. Hence it’s not the best thing for big datasets.

**Hence we do it differently using the Gaussian RBF Kernel** that allows us to still do the computation in the same dimensional space. If sigma is small, your circumference or boundary is small, if big, the circumference is bigger

1. **Naïve Bayes Classifier**
2. **Decision Trees:** CART – Classification and Regression Trees

**It** works off splits.

**We** change the criterion to ‘entropy’

1. **Confusion Matrix and Accuracy:** Accuracy Rate = TN + TP / Total

Error Rate = FP + FN / Total

1. **Accuracy Paradox:** completely abandoning your model and still getting a good accuracy hence accuracy alone as a model evaluation may not be efficient enough.
2. **Cumulative Accuracy Profile** is then used to get a better evaluation of the model.

**Perfect model, good model and random model**

X < 60 – Rubbish

60 < X < 70 – Poor

70 < X < 80 – Good

80 < X < 90 – Very Good

90 < X < 100 – Too Good : (**Note: may have overfitting)**

1. **ROC – Receiver Operating Characteristic**

**CLUSTERING**

1. **K- Means Clustering:** involves continuous determining first the number of clusters you want, if 2, randomly place 2 points on the dimension, then get a line that is equidistant to the points and then get the center of mass and shift the points to their appropriate center of mass and repeat the process accordingly.

**Elbow method:** pretty good method for determining the optimal number of clusters

**K-Means++** - used to escape the random initialization trap of the centroids.

**Here** we deploy a weighted random fashion of choosing a centroid hence reduces the change of random initialization trap.

**WCSS (Within-Cluster Sum of Square)** the sum of the squared distance between each point and the centroid in a cluster.

**The best number of clusters is the one where the WCSS starts reducing gradually.**