# Data Science

# Important Steps Before Training a Model

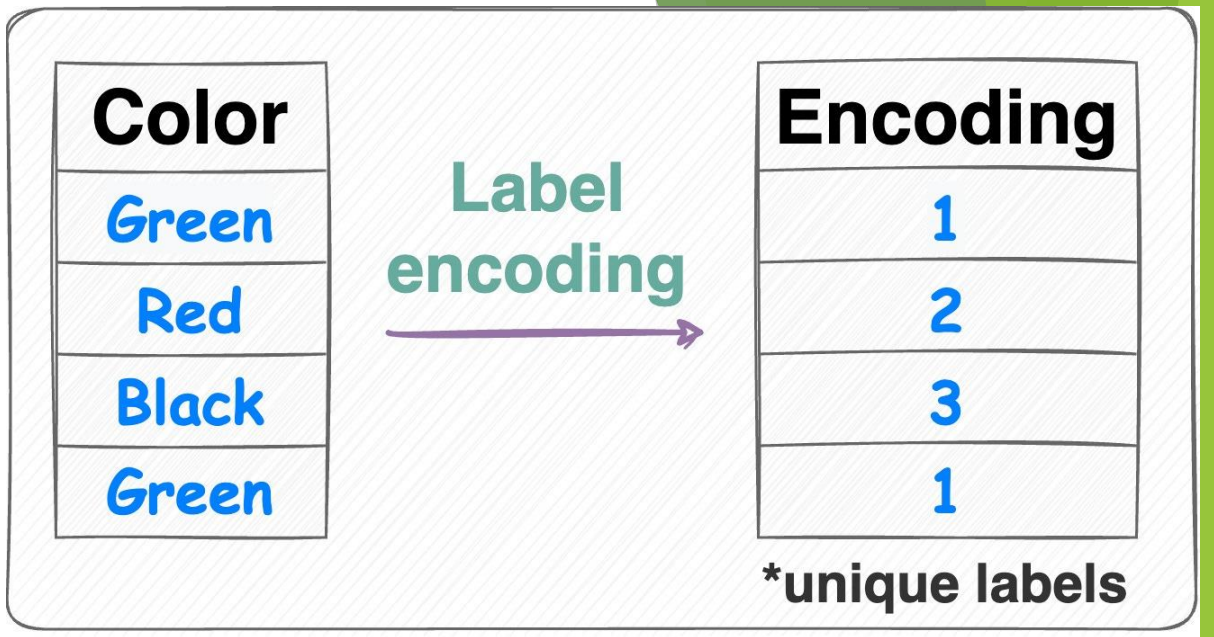
- **Handle Missing Values**:
  - For numerical columns: Use mean, median, or mode.
  - For categorical columns: Use the most frequent category.
  - We can drop the rows with missing values if missing values are very less
- **Handle Categorical/String Values**:
  - Replace categories with numerical representations (e.g., label encoding, one-hot encoding).
- **Handle Outliers**:
  - Identify using box plots or the empirical rule (68-95-99 rule) and remove them.
- **Remove Irrelevant Features**:
  - Use Exploratory Data Analysis (EDA) to identify and drop unnecessary features.
- **Split Data**:
  - Divide the dataset into training and testing sets (e.g., 80% train, 20% test).
- **Scale the Data**:
  - Apply techniques like standardization or normalization to ensure features are on a similar scale.

# Handling missing values

- Drop the columns with lot of missing values.
- Drop rows containing missing values. Ignore this if lot of rows are missing else data will get reduce.
- For numerical feature you can replace the missing value with either mean, median or mode. For eg if age is missing then you can use median or mode or if price is missing you can use mean.
- For categorical features try to replace it with most frequent value or add another category as ‘missing’ if missing value has meaning.
- From all above technique it depends on type of data which technique should be used.

# Handle categorical columns

- Label encoding: Here we replace the category with unique whole number.
- one hot encoding: Here we create a new column for each category and assign 1 if category is present else 0.
- one hot encoding increases number of features so it increases model training time significantly.
- When to use which? Usually one hot encoding works better compared to label encoding but if number of unique categories are more then it should be avoided.




| id | color |
|---|---|
| 1 | red |
| 2 | blue |
| 3 | green |
| 4 | blue |

One Hot Encoding

| id | color_red | color_blue | color_green |
|---|---|---|---|
| 1 | 1 | 0 | 0 |
| 2 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 |
| 4 | 0 | 1 | 0 |

# Handle text data (BOW)

- Here also we can use one hot encoding but on word level (Bag of words)
- Here first you find all words from all rows and create a column for each word and then use similar approach as was in one hot encoding.
- Can we very useful for spam email detection, sentiment analysis of movies, etc

| | about | bird | heard | is | the | word | you |
|---|---|---|---|---|---|---|---|
| About the bird, the bird, bird bird bird | 1 | 5 | 0 | 0 | 2 | 0 | 0 |
| You heard about the bird | 1 | 1 | 1 | 0 | 1 | 0 | 1 |
| The bird is the word | 0 | 1 | 0 | 1 | 2 | 1 | 0 |

# Handling Outliers

- Methos: Box plot or empirical rule (68-95-99) rule.
- We can either remove the outlier or can modify its value to nearest inlier value.
- Note that its not guaranteed that by removing outlier model performance increases but its usually based on choice whether to keep outliers or remove them.

# Remove Irrelevant Features:

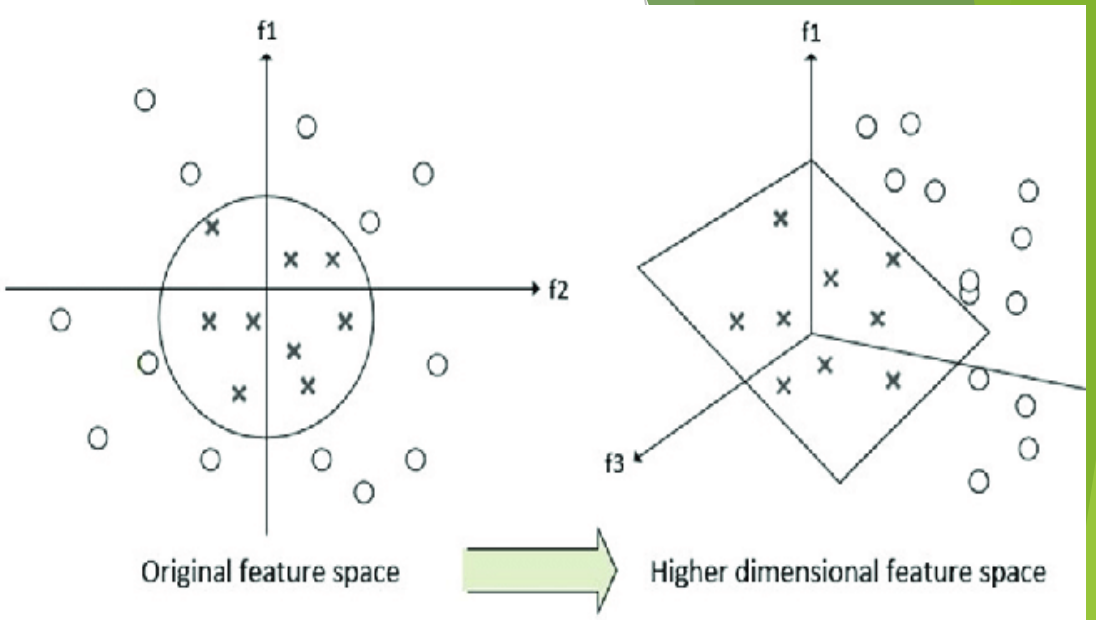
- We can perform indepth analysis to see the impact of each column with the prediction column.
- We can use pdf, scatter plot or can check correlation matrix and can use them to remove less important features.
- There is another way to use exhaustive search for which we give min_features and max_features to kept and it will one by one train model with all permutation combination of features and give best features but it is very expensive to run.
- Note that in real world no technique can determine if feature is important or not.

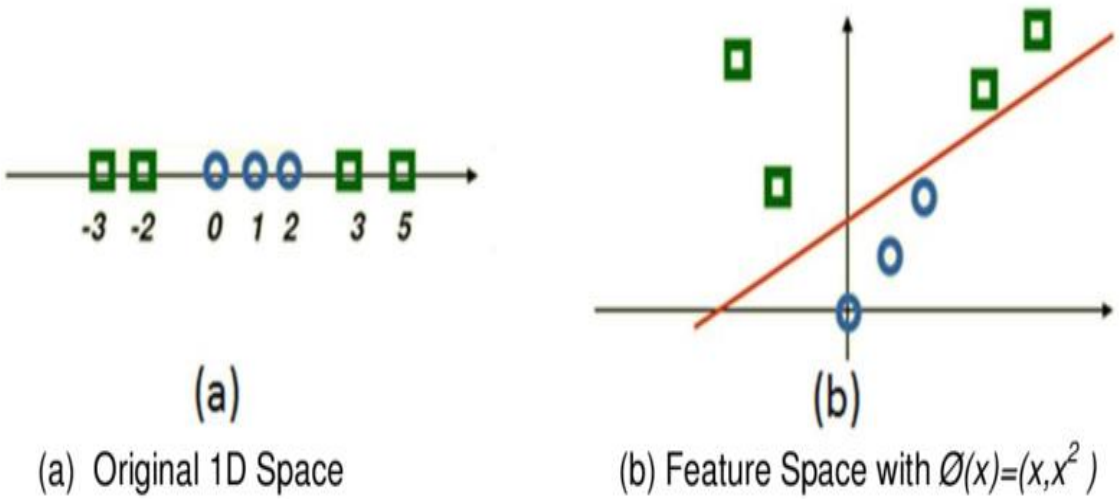
# Scaling of data

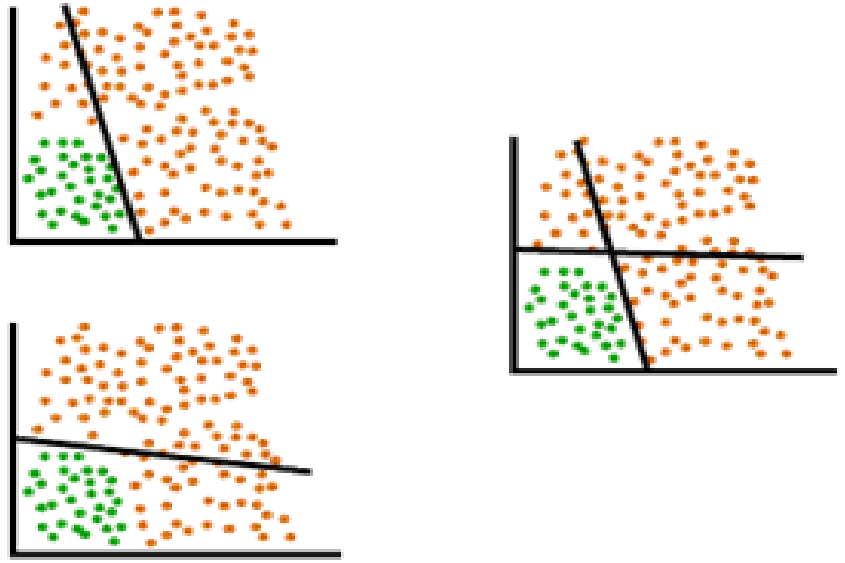
- Its very important to scale the data so that each column range is same.
- For algorithms where distance calculation is involved, scaling is very important else algorithm is biased towards certain columns. Eg knn, svm, etc
- For non distance based algorithms scaling is not needed. For eg decision tree, random forest, etc
- There are lot of techniques to scale data for eg normalization, standardization, robust scaling, etc. Here which one to use depends solely on type of data and problem.

# Feature engineering

- It basically means to create new features or modify existing features for optimal results.
- For eg if zip code is given then you can find distance of house from nearest station or airport and see its significance.
- Also you can perform some transformations on columns to convert non linear classification problem to linear classification problem.



# Splitting of data

- Its very important to randomly split your data into train and test.
- Usually train data is used to train the model to learn the pattern and whatever pattern model has learned from train data has to be tested on test data.
- Its very useful when you want to compare performance of lot of algorithms and find best one for production server.
- Never use training performance to find best model.
- Splitting of train and test data should be such way that training data should be given more and test data should be less. Typical ratios are (80-20, 70-30, 60-40). Note it also depends on number of data points to decide the splitting ratio.

# Classification metrics

- Most common metric for classification is accuracy score.
- Its is defined as number of correct prediction divided by total number of predictions.
- Accuracy should be always greater then 50 % because if its around 50 then it means that model prediction is solely based on random or chance.
- There are other metrics as well like precision, recall, roc-auc, etc
- Type of metrics to check the performance or find best model out of lot of models is based on problem statement and what you are solving.

# Real world scenerios

- Its very difficult to get very good metrics for eg 90+ accuracy because real world scenario is quite complex. Accuracy should be always greater than 60% else you can say model is random. Also if you are getting more than 90% accuracy recheck it once again.
- Don’t just look at performance also look as cost, time for prediction, model complexity, etc to finalize the best model for production.
- Note that everything depends on data and not the model. Model is just replica of data it has been trained on. So proper EDA and preprocessing should be performed in depth to find hidden patterns in data. So instead of trying out more and more complex models try to find hidden pattern in your data.
- Sometimes it may happen that you tried everything still you are not getting optimal results so try to see if there is any pattern in your data or not because not all the data can be used for training and prediction.