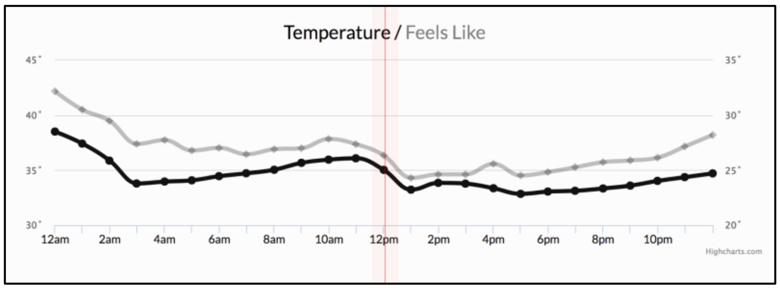
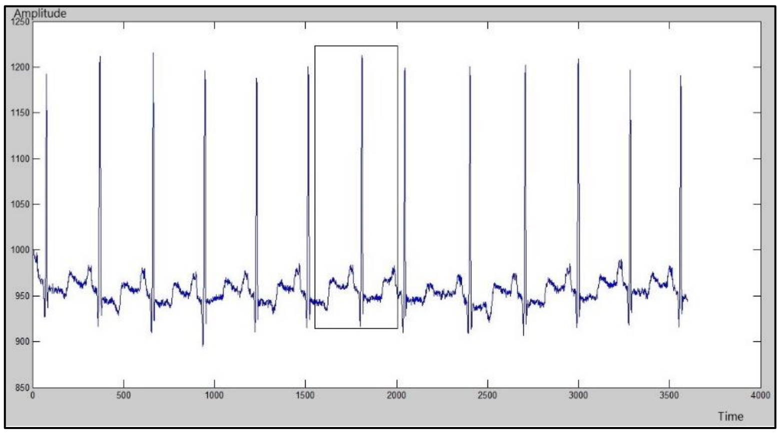
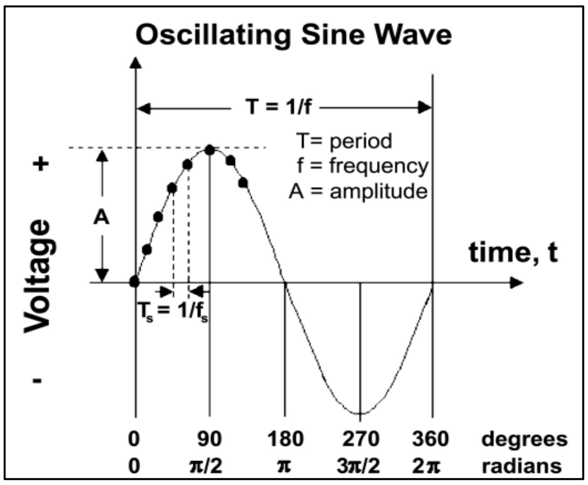
- **Introduction**
  - Most important process in ML is feature engineering; Data can be in the form of Text, categorical, numerical, time series, image, database tables, Graph-data, etc.
  - Text data: BOW, TFIDF, AVGW2V, TFIDFW2V
  - Categorical data: One-hot Encoding, response encoding
  - Time series e.g., heart rate, stock market data: transformed to numerical vector, frequency domain, etc.



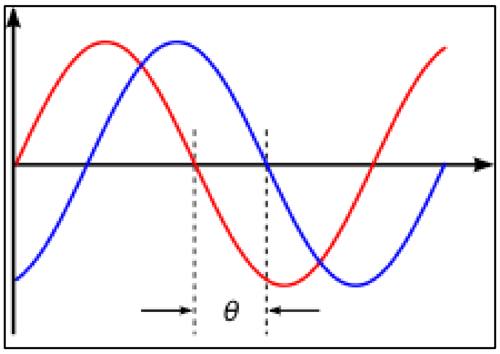
  - Image data e.g., face detection: transformed to arrays
- **Moving window for Time Series Data**
  - Simplest featurization of time series data



  - The above figure represents time on x axis and amplitude on y axis
  - Moving window takes a time split of a width of time
  - Some standard features extracted from window of the time series are Mean, Median, Max and min, Max - min, Standard deviation, quantile, number of local minima or maxima, mean crossings or zero crossings all of this for each window
- **Fourier Decomposition**

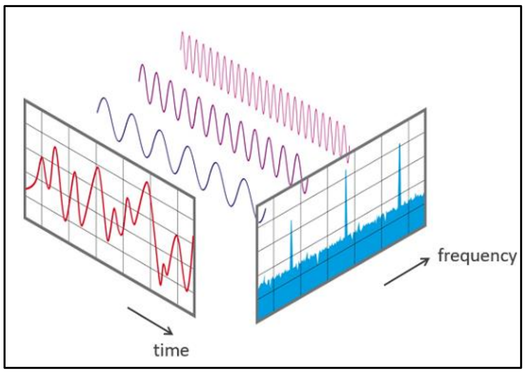


  - A Method to represent time series data
  - T = Time taken for each oscillation
  - Freq = 1/T (Hz) e.g., Heart rate: 60 – 100 BPM = 1 – 1.6 BPS = 1 – 1.6 Hz
  - Phase: Difference in angle at which different waves start



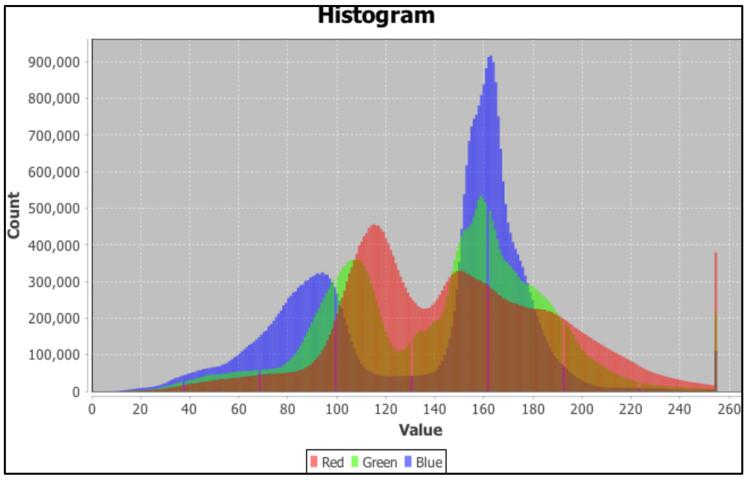
  - Just like heart rates, repeating behaviours also occurs in ecommerce website. E.g., We can have daily shopping patterns or weekly shopping patterns or annual patterns
  - Given a composite wave: It can be decomposed into multiple sine waves, different sine waves have different time period, and different peaks: this is transformed onto frequency domain; This process is called as Fourier transformation

- Representing/converting more complex waveform into simple individual waves so that we can extract most information out of it. E.g., Indian curry

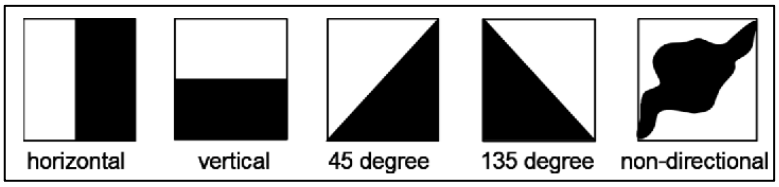


- In order to analyse (amount of ingredients) the prepared curry is so difficult i.e., ingredients like amount of salt, Turmeric powder, chilli powder and type of masala etc.
- So, we pass the prepared curry through a complex method called Fourier transforms so that we get most individual ingredients out of prepared curry.
- Note that ingredients obtained is sometimes in its initial/pure stage or sometimes impure. But we can analyse individual ingredients better than from the processed curry.
- After passing the curry through F.T we get ingredients individually and to re-obtain the curry back we pass it through Inverse Fourier transforms.
- In mathematical point of view curry is any complex signal like audio /Video etc and ingredients refers to frequency, amplitude and phase.

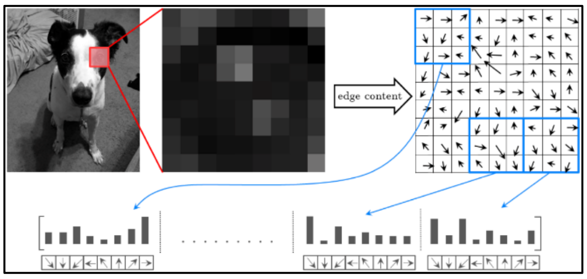
- **Deep Learning Feature: LSTM**
  - For each problem: a specific set of features were designed from domain expertise. A set of features designed for a problem may not work for another problem.
  - Deep learning automatically learns the best features from lot of data.
  - Deep Learnt Features: Used for time series using LSTM. E.g., google now, Siri, song recommendations, etc.
- **Image Histogram**
  - Colour & edge histogram are very rudimentary & basic object recognition techniques
  - **Colour histogram**: Each pixel has 3 Colour values i.e., RGB. For each Colour plot a histogram (0 to 255) and vectorize it
  - Different objects have distinct Colour attributes, sky and water come in blue, human faces have cream, wood brown, metal grey, etc.
  - Colour histogram cannot recognise shapes



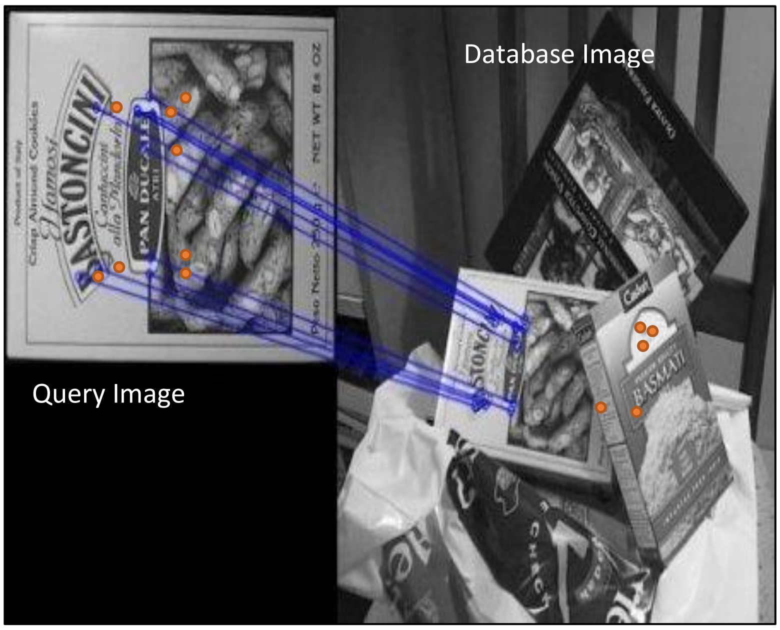
- **Edge histogram**:
  - At the interface of Colour separation, we detect region-based edge orientation
  - These edges are selected based on angle



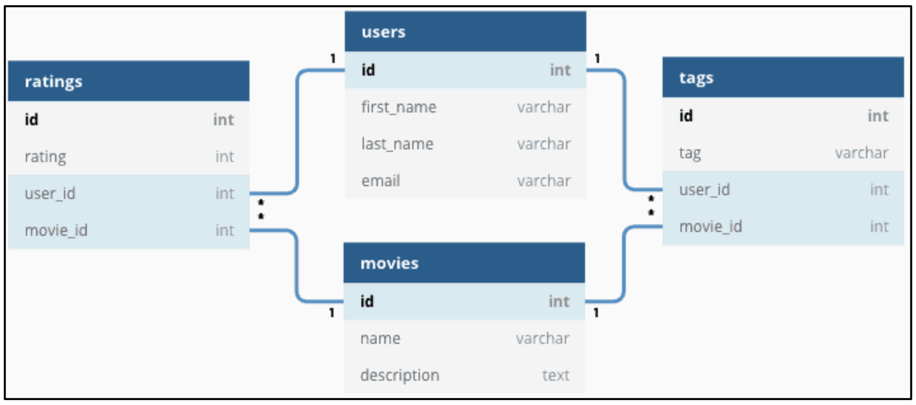
  - First the image is divided into square segments. For each segment the angle is calculated.



- Key points: SIFT (Scale Invariant Feature Transforms)

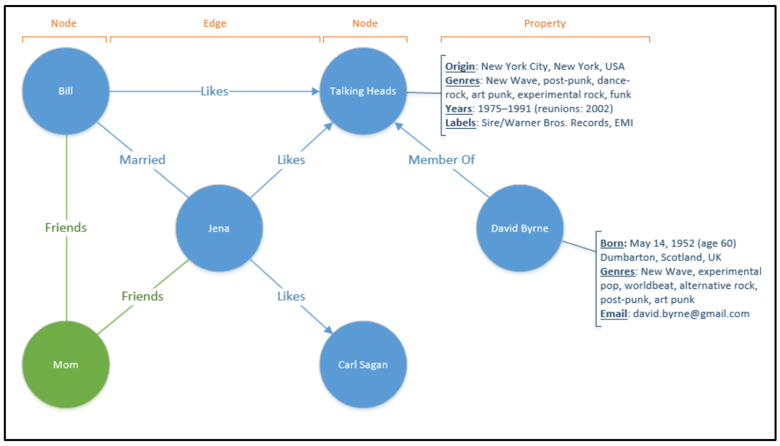


  - When an image is processed through SIFT, it aims to find highly-distinctive interest points (or key points) in an image.
    1. First, we'd need blur and resample the image with different blur widths and sampling rates to create a scale space
    2. Use the difference of gaussians method to detect blobs at different scales; the blob centres become our key points at a given x, y, and scale (That is why in general the key points are the corners of an image where the gradient w.r.t to x and y is high)
    3. Assign every key point a 128-dimensional feature vector based on the gradient orientations of pixels in 16 local neighbourhoods.
  - For each of this key point, there is a 128-dimensional vector
  - When an image is searched, all key points are detected by SIFT. Same key points are checked in database image.
  - Wherever a set of this key points matches we can say that it's a match
  - SIFT is scale & rotation invariant as features of the image will be same even if image is smaller or bigger (scale) or even if the image is rotated as SIFT detects the corner points only.
  - OpenCV: Feature Matching
- **Deep learning features: Convolutional neural network (CNN)**
  - Just like LSTM was for time series data. CNN is for image processing
- **Relational data**



  - Relational data is a Data that is stored in multiple tables and connected through relations
  - Typically, relational data is by using SQL databases. E.g., Oracle, MySQL, SQL Server

- **Graph Data**
  - A graph is a mathematical construct consisting of a set of objects (nodes) and the connections between these objects (edges). E.g., for a Facebook objects will be users & connection will be the friendship



Node
Edge
Node
Property
Bill
Likes
Talking Heads
Origin: New York City, New York, USA
Genres: New Wave, post-punk, dance-rock, art punk, experimental rock, funk
Years: 1975–1991 (reunions: 2002)
Labels: Sire/Warner Bros. Records, EMI
Married
Likes
Member Of
Friends
Jena
David Byrne
Born: May 14, 1952 (age 60) Dumbarton, Scotland, UK
Genres: New Wave, experimental pop, worldbeat, alternative rock, post-punk, art punk
Email: david.byrne@gmail.com
Friends
Likes
Mom
Carl Sagan

  - Example: Facebook: User Social Graph
    - Task: Recommend new friends
    - Features: Number of paths different nodes of the Graph, Number of mutual friends
- **Indicator variables**
  - Converting a feature into indicator variables depending on a problem we are facing
    - Real valued Feature if Height > 150, 0 else 1
    - Categorical Feature If country == India Or USA return 1 else 0
- **Feature binning**
  - It is a logical extension to indicator variable
  - If H < 120

```
If H < 120
          return 1
elif H ≥ 120 & H < 150
          return 2
elif H ≥ 150 & H < 180
          return3
else
          return 4
```

  - Each of this condition is a bin or bucket such that we are dividing whole data in 4 categories
  - Selecting the threshold values is very problem specific
    - Task: Predict gender given, height, weight, hair length, eye color: return 1 if male and return 0 if female
    - Train a simple decision tree using height which will give us the appropriate threshold value to categorise the data further
  - If we have to convert real valued features into binning Decision Tree Based Feature Binning is the best way
- **Interaction Variables**
  - 1. if h<150 AND w<=60 kgs return 0: Logical 2-way interaction
  - 2. h * w OR w*h2: Mathematical 2-way interaction here
  - 3. h < 150 AND w<65 AND hair length > 75 return 0: Logical 3-way interaction
- **Mathematical transformations**
  - If X is a feature, then different mathematical operations $\log(X)$, e^x Trigonometrical (sin, cos, tan), polynomial (X^n), $n\sqrt{x}$ etc. can be used to transform the features;
  - E.g., we had converted power law distribution to gaussian distribution by using box cox transform which was by using log(x)
- **Model specific featurization's**
  - Say f1 is power law distributed: when using Logistic Regression: transforming with log is beneficial;
  - 3 features and a regression target: say we have Y ≈f1 – f2 + 2 f3: Linear models are better; Decision Trees may not work here
  - If interaction of features exists, DT perform well
  - With Bag of Words: Linear models tend to perform very well (Lr SVM and Logistic Regression)

- **Feature orthogonality**
    - The more different/orthogonal the features (Orthogonal relationship) are, the better the models will be
    - If there are correlations (less orthogonality) among the features, the overall prediction performance of the model on the target will be poor
    - If there are no correlation amongst the features but these features are correlated to the y, this will give better prediction result
    - This implies, if we are adding a new feature to the dataset it should be correlated to the y but very less correlated to existing features.
    - These new features shall be added such that they are correlated to the errors we are getting from previous trained model like boosting
    - But it can also easily overfit so it needs to be designed carefully
- **Feature slicing**
    - If there is dominant categorical feature present in database. The model will perform better on more dominant feature as compared to the less dominant one.
    - To prevent this from happening we split the data into two parts according to this features and train them separately
    - We will check the error distribution on the categorical data if those two error distributions are different from each other then we will go for a splitting of data based on that categorical column and we will train two models.
    - Split criteria: feature values are different and sufficient data points are available for each category
    - Example: Customers buying Desktops and Mobile
- **Kaggle Winner Solutions**
    - Feature engineering methods can be learnt from Kaggle winner solutions, And discussion of competitions