Machine learning/ Natural language processing/ Information Visualization

Vimala Nunavath

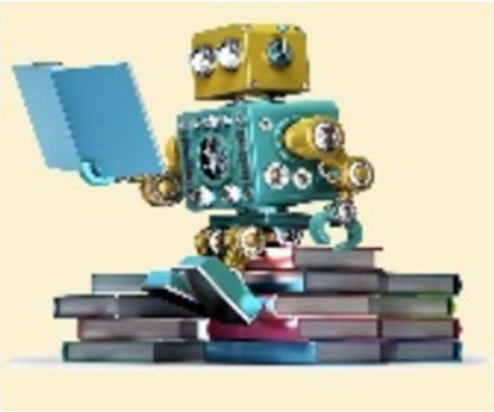
Vimala.Nunavath@uia.no

Agenda

- Machine learning
- Natural language processing
- Data visualization / Dashboards

Machine learning

Machine learning



"Programming computer to optimize the performance using example data and past experience"

Machine learning

Field of study that gives “computers the ability to learn without being explicitly programmed”

---- Arthur Samuel, 1959

Machine learning

- Branch of Artificial Intelligence
- Design and development of algorithms
- Computers evolve behavior based on empirical data

Machine learning

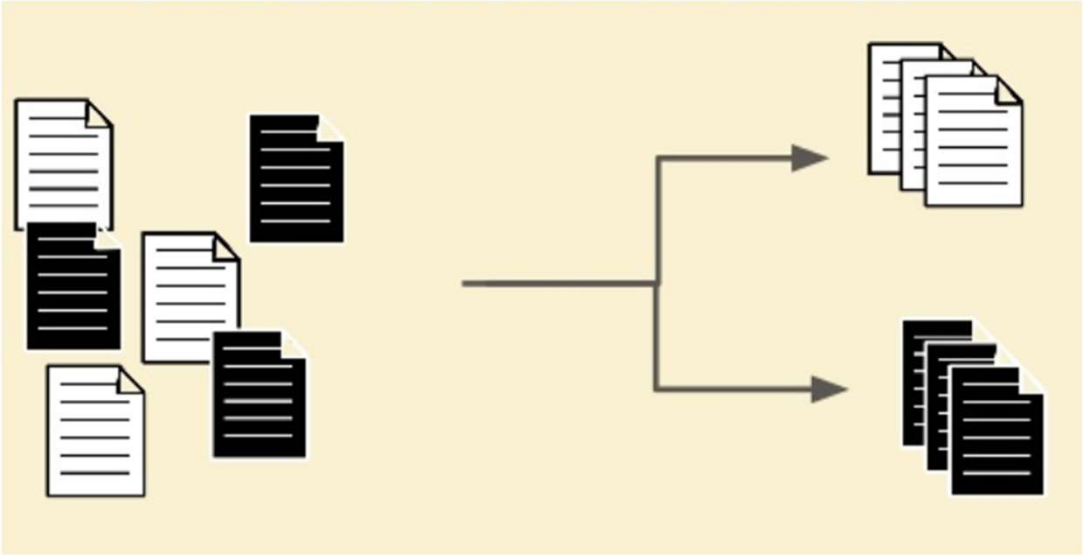
- The input data to a machine learning system can be numerical, textual, audio, visual, or multimedia.
- The corresponding output data of the system can be a floating-point number, for instance, the velocity of a rocket, an integer representing a category or a class, for example, a pigeon or a sunflower from image recognition.

Machine learning - Applications

- Recommend friends, dates and products to end-users.

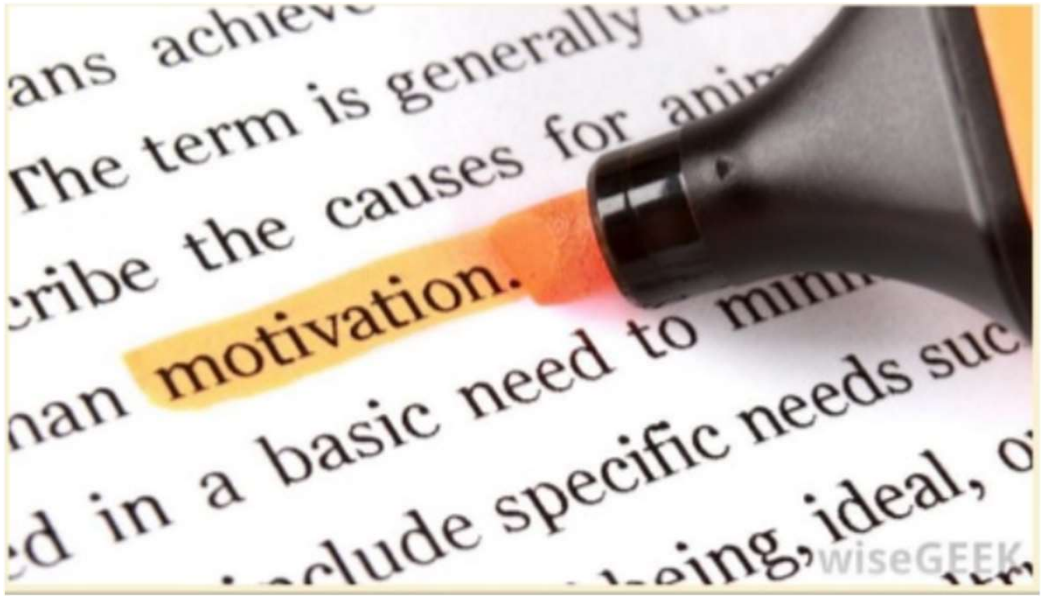
Machine learning - Applications

- Classify content into predefined groups



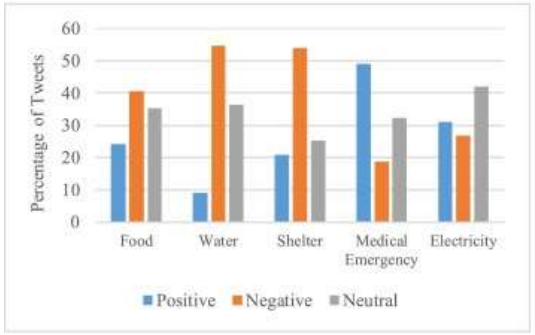
Machine learning - Applications

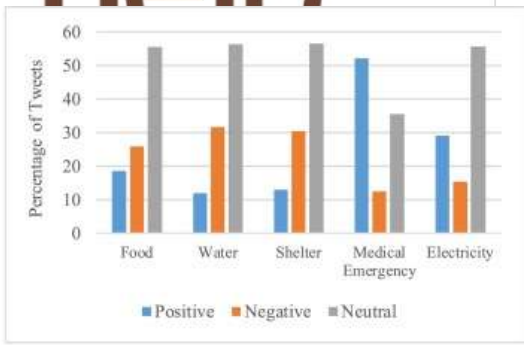
- Identify key topics in large collections of text



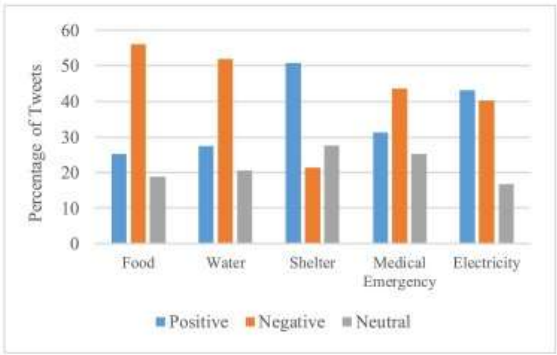
Machine learning - Applications

- Natural language processing





(a) (b)

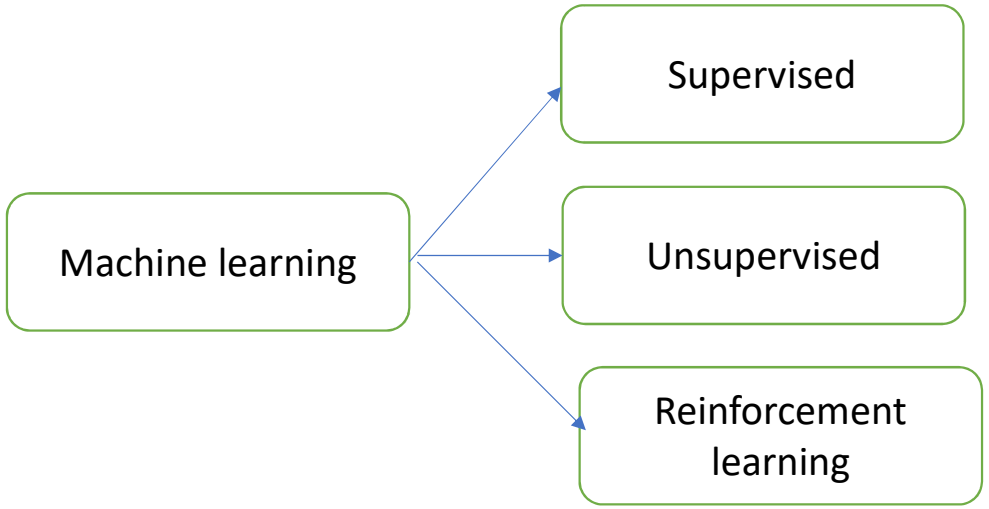


(c)

Machine learning - Applications

- Find Similar content based on Object Properties.
- Detect Anomalies within given data.
- Ranking Search Results with User Feedback Learning
- Classifying DNA sequences.
- Sentiment Analysis/ Opinion Mining
- BioInformatics.
- Speech and HandWriting Recognition.

Machine learning types

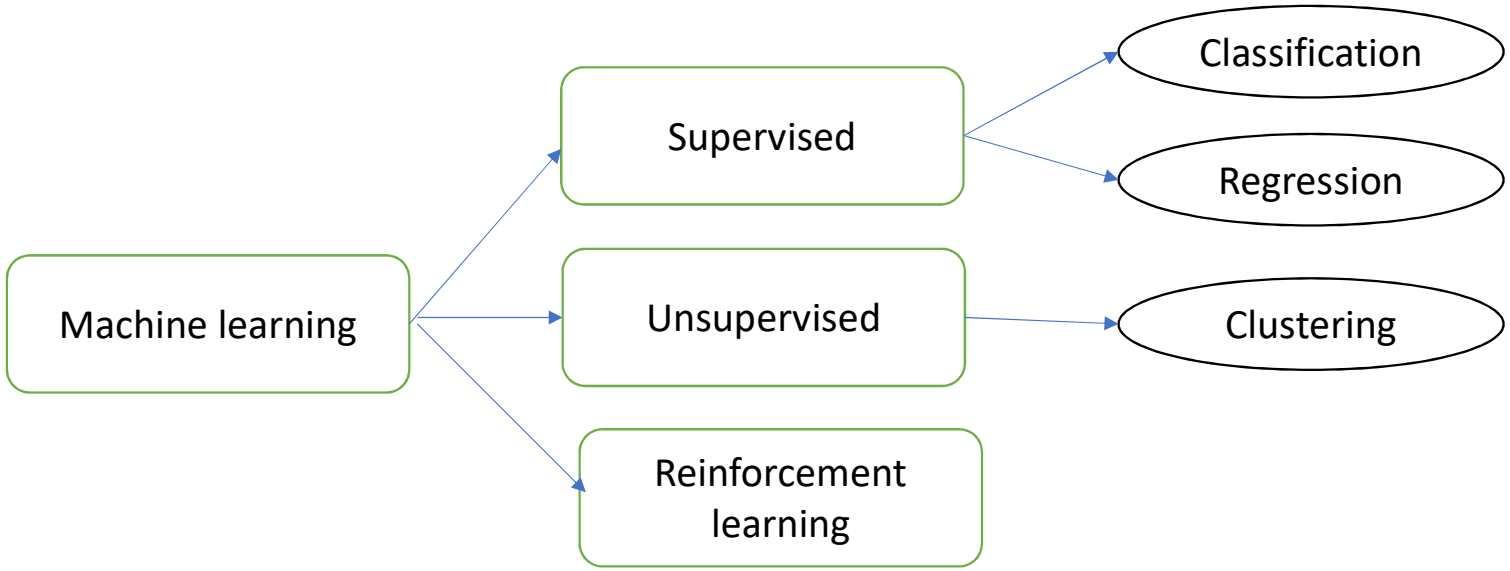


Given example inputs and outputs,
Learn to map inputs to outputs

No labels are given, find structure

Dynamic environment, perform a certain goal

Machine learning types



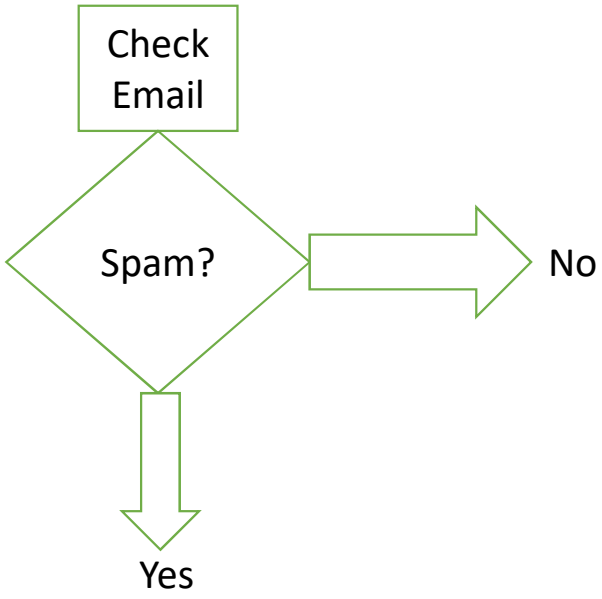
Machine learning types - Supervised learning

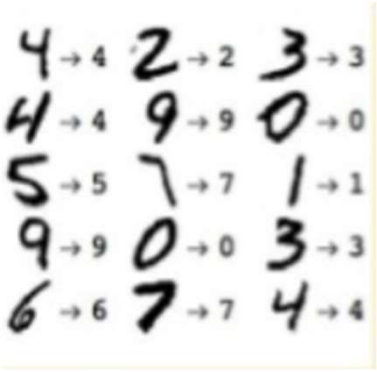
- Supervised learning deals with labeled training data.
- It is broadly categorized into two major types.
  - Classification
  - Regression
- Classification: It predicts categorical variables and classes.
- The classification algorithms are: Naïve Bayes, Decision Trees, and Ensembles of trees (Random Forests and Gradient-Boosted Trees)
- Regression: It deals with a target variables and is continuous.
- The regression algorithms are: Linear regression, logistic regression and SVM.

Machine learning types - Unsupervised learning

- Unsupervised learning deals with unlabeled data.
- The objective is to observe structure in the data and find patterns.
- Tasks such as cluster analysis, association rule mining, outlier detection, dimensionality reduction can be modeled as unsupervised learning problems.
- K-means: This is the task of grouping similar objects (called a cluster) together to partition n observations into k clusters, for example, clustering similar tweets, topics and same area, determine earthquake danger zones.

Machine learning - Classification



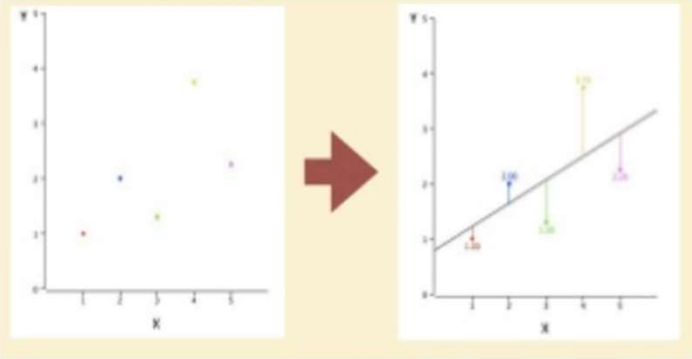


Machine learning - Regression

Table 1. Example data.

X	Y
1.00	1.00
2.00	2.00
3.00	1.30
4.00	3.75
5.00	2.25

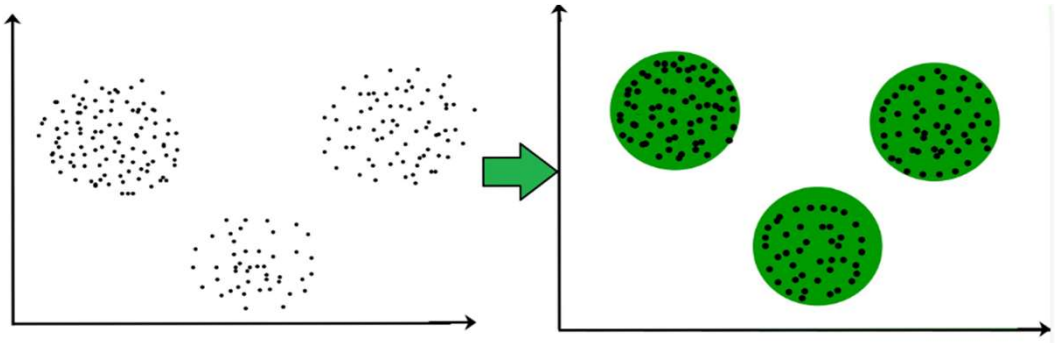
Predicting a continuous –valued attribute associated with an object



In liner regression, we draw all possible lines going through the points such that it is closest to all

Machine learning - Clustering

- **Clustering** is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups.
- It is basically a collection of objects on the basis of similarity and dissimilarity between them.

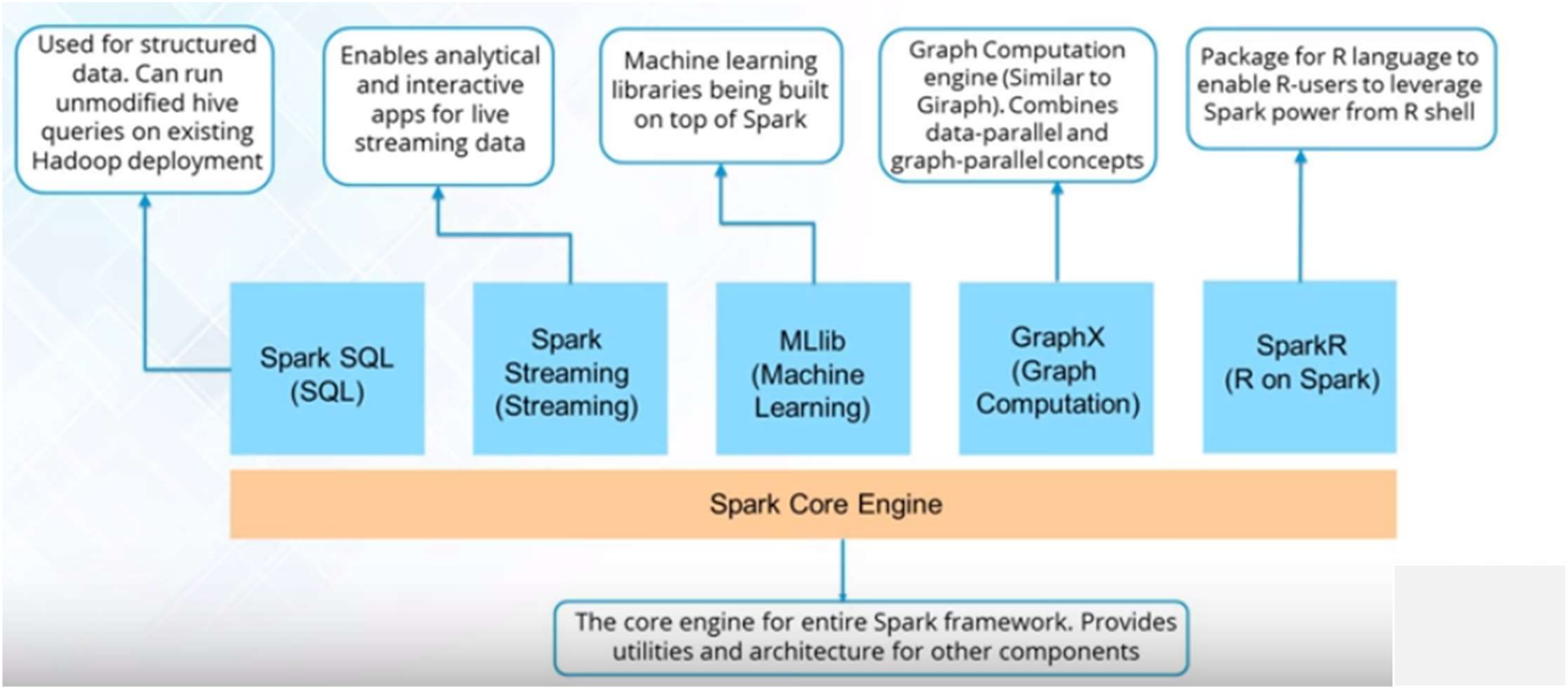


Machine learning - Tools

DATA SIZE	CLASSFICATION	TOOLS
Lines Sample Data	Analysis and Visualization	Whiteboard,…
KBs - low MBs Prototype Data	Analysis and Visualization	Matlab, Octave, R, Processing,
MBs - low GBs Online Data	Analysis	NumPy, SciPy, Weka,
	Visualization	Flare, AmCharts, Raphael, Protovis
GBs - TBs - PBs Big Data	Analysis	MLlib, SparkR, GraphX, Mahout, Giraph

Spark MLlib

Spark ecosystem



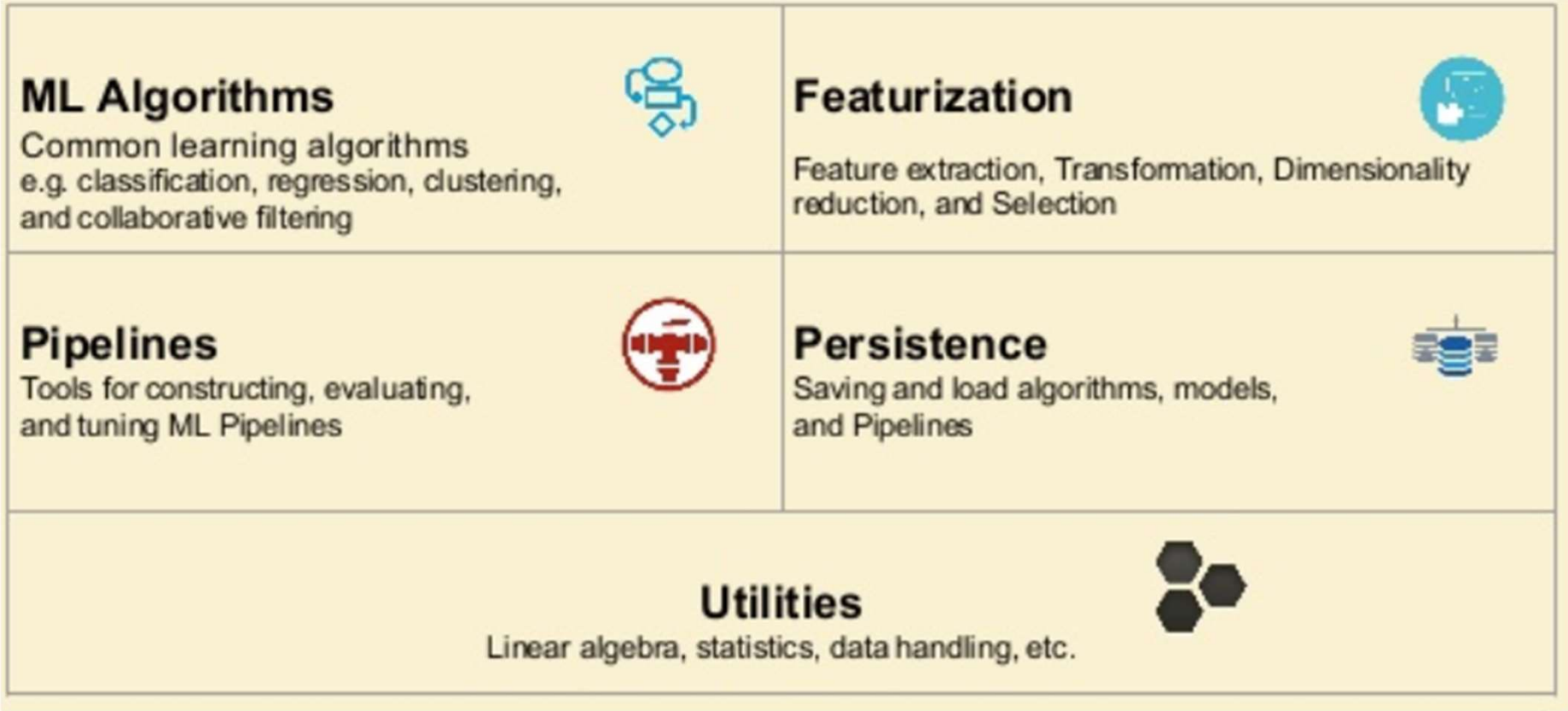
Spark MLlib Overview

- Spark MLlib is used to perform machine learning in Apache Spark. MLlib consists popular algorithms and utilities.

  - *spark.mllib* contains the original API built on top of RDDs. It is currently in maintenance mode.

  - *spark.ml* provides higher level API built on top of DataFrames for constructing ML pipelines. It is the primary Machine Learning API for Spark at the moment.

MLlib Structure



MLlib Algorithms - Clustering

- Clustering is the task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar (in some sense or another) to each other than to those in other groups (clusters).
- So, it is the main task of exploratory data mining, and a common technique for statistical data analysis, used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, bioinformatics, data compression and computer graphics.

Spark.mllib - Data types

Local vector

integer-typed and 0-based indices and double-typed values

dv2 = [1.0, 0.0, 3.0]

Labeled point

a local vector, either dense or sparse, associated with a label/response

pos = LabeledPoint(1.0, [1.0, 0.0, 3.0])

Matrices:

Local matrix

Distributed matrix

RowMatrix

IndexedRowMatrix

CoordinateMatrix

BlockMatrix

Main concepts of MLlib

- **ML Dataset:** Spark ML uses the DataFrame from Spark SQL as a dataset which can hold a variety of data types. E.g., a dataset could have different columns storing text, feature vectors, true labels, and predictions.
- **Transformer:** A Transformer is an algorithm which can transform one DataFrame into another DataFrame. E.g., an ML model is a Transformer which transforms an RDD with features into an RDD with predictions.
- **Estimator:** An Estimator is an algorithm which can be fit on a DataFrame to produce a Transformer. E.g., a learning algorithm is an Estimator which trains on a dataset and produces a model.
- **Pipeline:** A Pipeline chains multiple Transformers and Estimators together to specify an ML workflow.
- **Param:** All Transformers and Estimators now share a common API for specifying parameters

Concepts of MLlib - ML Dataset

- Machine learning can be applied to a wide variety of data types, such as vectors, text, images, and structured data. Spark ML adopts the DataFrame from Spark SQL in order to support a variety of data types under a unified Dataset concept.
- DataFrame supports many basic and structured types; see the Spark SQL datatype reference for a list of supported types. In addition to the types listed in the Spark SQL guide, DataFrame can use ML Vector types.
- A DataFrame can be created either implicitly or explicitly from a regular RDD. See the code examples below and the Spark SQL programming guide for examples.
- Columns in a DataFrame are named. The code examples below use names such as “text,” “features,” and “label.”

ML Algorithms - Transformers

- A Transformer is an abstraction which includes feature transformers and learned models.
- Technically, a Transformer implements a method transform() which converts one DataFrame into another, generally by appending one or more columns.
- For example: A feature transformer might take a dataset, read a column (e.g., text), convert it into a new column (e.g., feature vectors), append the new column to the dataset, and output the updated dataset.
- A learning model might take a dataset, read the column containing feature vectors, predict the label for each feature vector, append the labels as a new column, and output the updated dataset.

Main concepts of MLlib - Estimator

- An Estimator abstracts the concept of a learning algorithm or any algorithm which fits or trains on data.
- Technically, an Estimator implements a method fit() which accepts a DataFrame and produces a Transformer.
- For example, a learning algorithm such as LogisticRegression is an Estimator, and calling fit() trains a LogisticRegressionModel, which is a Transformer

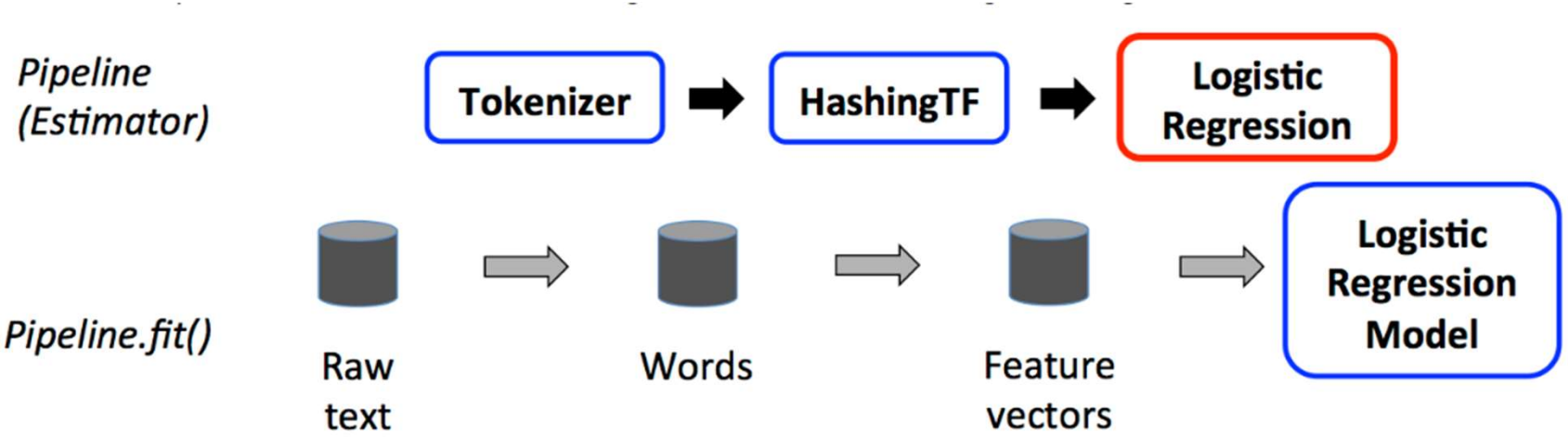
Pipeline

- In machine learning, it is common to run a sequence of algorithms to process and learn from data.
- E.g., a simple text document processing workflow might include several stages:
- - Split each document’s text into words.
- - Convert each document’s words into a numerical feature vector.

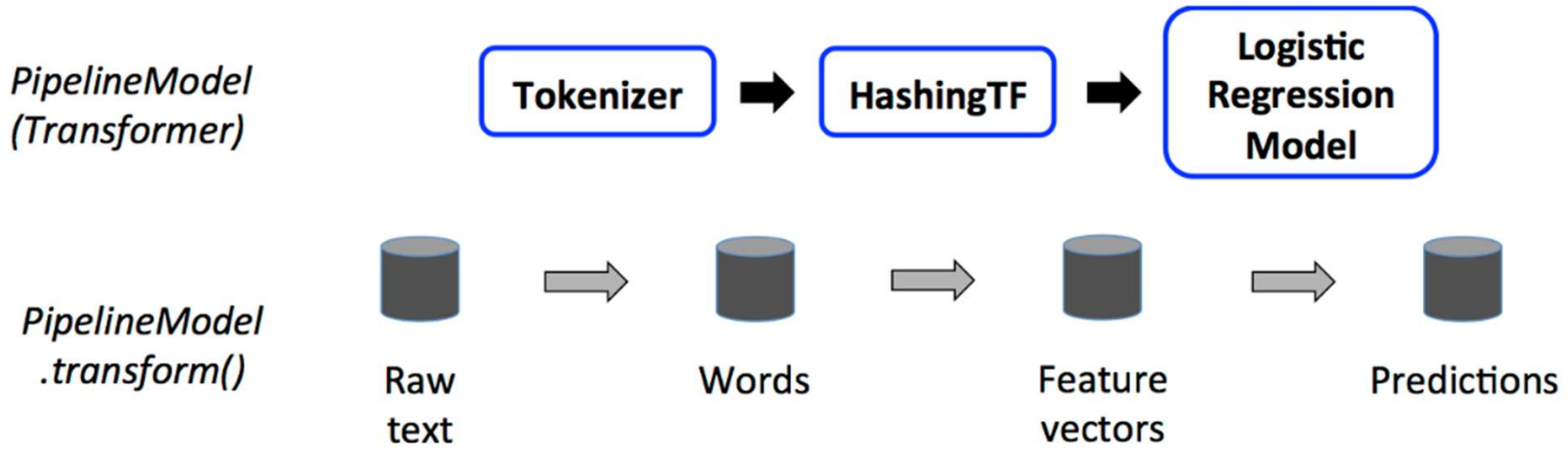
- Learn a prediction model using the feature vectors and labels.

- Spark ML represents such a workflow as a Pipeline, which consists of a sequence of PipelineStages (Transformers and Estimators) to be run in a specific order.

Pip Lines – how it works?



Pip Lines – how it works?



Parameters

- Spark ML Estimators and Transformers use a uniform API for specifying parameters.
- A Param is a named parameter with self-contained documentation.
- A ParamMap is a set of (parameter, value) pairs.
- There are two main ways to pass parameters to an algorithm:

  - Set parameters for an instance.

  E.g., if lr is an instance of LogisticRegression, one could call lr.setMaxIter(10) to make lr.fit() use at most 10 iterations. This API resembles the API used in MLlib.

  - Pass a ParamMap to fit() or transform(). Any parameters in the ParamMap will override parameters previously specified via setter methods.

- Parameters belong to specific instances of Estimators and Transformers.

  - For example, if we have two LogisticRegression instances lr1 and lr2, then we can build a ParamMap with both maxIter parameters specified: ParamMap(lr1.maxIter -> 10, lr2.maxIter -> 20). This is useful if there are two algorithms with the maxIter parameter in a Pipeline.

Machine learning algorithms

Type of machine learning	Type of algorithm	Algorithm name
Supervised learning	Classification	Naïve Bayes
		Decision Trees
		Random Forests
		Gradient-Boosted Trees
	Regression	Linear Regression
		Logistic Regression
		Support vector machines
Unsupervised learning	Clustering	K-means
		Gaussian Mixture
		Power Iteration Clustering (PCA)
		Latent Dirichlet Allocation (LDA)
		Streaming K-means
	Dimensionality reduction	Singular Value Decomposition (SVD)
		Principal Component Analysis (PCA)
Recommended Systems	Collaborative filtering	User-based collaborative filtering
		Item-based collaborative filtering
		Alternating Least Squares (ALS)
Feature Extraction	Feature extraction and transformation	TF-IDF
		Word2Vec
		Standard Scaler
		Normalizer
		Chi-square Selector
Optimization	Optimization	Stochastic Gradient Descent
		Limited-memory BFGS

Feature Extraction and transformation

- Feature extraction and transformation are essential techniques to process large text documents and other data sets.
- It contains 5 techniques:
  - **Term frequency (TF) and inverse document frequency (IDF)**
  - **Word2Vec**
  - Standard scaler
  - Normalizer
  - Chi-square Selector

Term frequency (TF) and inverse document frequency (IDF)

```
from pyspark import SparkContext
from pyspark.mllib.feature import HashingTF

sc = SparkContext()

# Load documents (one per line).
documents = sc.textFile("...").map(lambda line: line.split(" "))

hashingTF = HashingTF()
tf = hashingTF.transform(documents)
```

```
# ... continue from the previous example
tf.cache()
idf = IDF(minDocFreq=2).fit(tf)
tfidf = idf.transform(tf)
```

```
from pyspark.mllib.feature import IDF

# ... continue from the previous example
tf.cache()
idf = IDF().fit(tf)
tfidf = idf.transform(tf)
```

Word2Vec

```
from pyspark import SparkContext
from pyspark.mllib.feature import Word2Vec

sc = SparkContext(appName='Word2Vec')
inp = sc.textFile("text8_lines").map(lambda row: row.split(" "))

word2vec = Word2Vec()
model = word2vec.fit(inp)

synonyms = model.findSynonyms('china', 40)

for word, cosine_distance in synonyms:
    print "{}: {}".format(word, cosine_distance)
```

MLlib Algorithms

- The popular algorithms and utilities in Spark MLlib are:

1. Basic Statistics
2. Regression
3. Classification
4. Recommendation System
5. Clustering
6. Dimensionality Reduction
7. Feature Extraction

MLlib Algorithms - Basic Statistics

- Basic Statistics includes the most basic of machine learning techniques. These include:
  1. **Summary Statistics**: Examples include mean, variance, count, max, min and numNonZeros.
  2. **Correlations**: Spearman and Pearson are some ways to find correlation.
  3. **Stratified Sampling**: These include sampleBykey and sampleByKeyExact.
  4. **Hypothesis Testing**: Pearson's chi-squared test is an example of hypothesis testing.
  5. **Random Data Generation**: RandomRDDs, Normal and Poisson are used to generate random data.

MLlib Algorithms - Regression

- *Regression* analysis is a statistical process for estimating the relationships among variables.
- It includes many techniques for modeling and analyzing several variables when the focus is on the relationship between a dependent variable and one or more independent variables.
- More specifically, regression analysis helps one understand how the typical value of the dependent variable changes when any one of the independent variables is varied, while the other independent variables are held fixed.
- Regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning.
- Regression analysis is also used to understand which among the independent variables are related to the dependent variable, and to explore the forms of these relationships. In restricted circumstances, regression analysis can be used to infer causal relationships between the independent and dependent variables.

MLlib Algorithms - Classification

- Classification is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. It is an example of pattern recognition.
- Here, an example would be assigning a given email into “spam” or “non-spam” classes or assigning a diagnosis to a given patient as described by observed characteristics of the patient (gender, blood pressure, presence or absence of certain symptoms, etc.).

MLlib Algorithms - Recommendation System

- A recommendation system is a subclass of information filtering system that seeks to predict the "rating" or "preference" that a user would give to an item.
- Recommender systems have become increasingly popular in recent years, and are utilized in a variety of areas including movies, music, news, books, research articles, search queries, social tags, and products in general.
- Recommender systems typically produce a list of recommendations in one of two ways – through collaborative and content-based filtering or the personality-based approach.
    1. **Collaborative Filtering** approaches building a model from a user's past behavior (items previously purchased or selected and/or numerical ratings given to those items) as well as similar decisions made by other users. This model is then used to predict items (or ratings for items) that the user may have an interest in.
    2. **Content-Based Filtering** approaches utilize a series of discrete characteristics of an item in order to recommend additional items with similar properties.
- Further, these approaches are often combined as Hybrid Recommender Systems.

MLlib Algorithms - Dimensionality Reduction

- Dimensionality Reduction is the process of reducing the number of random variables under consideration, via obtaining a set of principal variables. It can be divided into feature selection and feature extraction.
    1. **Feature Selection:** Feature selection finds a subset of the original variables (also called features or attributes).
    2. **Feature Extraction:** This transforms the data in the high-dimensional space to a space of fewer dimensions. The data transformation may be linear, as in Principal Component Analysis(PCA), but many nonlinear dimensionality reduction techniques also exist.

MLlib Algorithms - Feature Extraction

- Feature Extraction starts from an initial set of measured data and builds derived values (features) intended to be informative and non-redundant, facilitating the subsequent learning and generalization steps, and in some cases leading to better human interpretations.
- Algorithms for working with features, roughly divided into these groups:
  1. Extraction: Extracting features from “raw” data
  2. Transformation: Scaling, converting, or modifying features
  3. Selection: Selecting a subset from a larger set of features
  4. Locality Sensitive Hashing (LSH): This class of algorithms combines aspects of feature transformation with other algorithms.

MLlib Algorithms - Optimization

- Optimization is the selection of the best element (with regard to some criterion) from some set of available alternatives.
- In the simplest case, an optimization problem consists of maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function.
- The generalization of optimization theory and techniques to other formulations comprises a large area of applied mathematics.
- More generally, optimization includes finding “best available” values of some objective function given a defined domain (or input), including a variety of different types of objective functions and different types of domains.

NLP

What is Natural language processing

- Natural Language Processing (NLP) is an area of research and application that explores how computers can be used to understand and manipulate natural language text or speech to do useful things.

The goal of NLP

- The goal of NLP as stated above is “to accomplish human-like language processing”.

- A full NLP System would be able to:
    1. Paraphrase an input text
    2. Translate the text into another language
    3. Answer questions about the contents of the text
    4. Draw inferences from the text

Why NLP?

- Aids communication between two humans
  - Machine translation
  - Speech-to-speech translation
  - Speech-to-text & text-to-speech
  - Editorial aids (spelling & grammar checkers)
  - Aids communication between human and machine
    - Personal assistants
    - Interactive Voice Response systems
  - Aids communication between two machines

Why NLP for Social Media?

- Social Media generates BIG UNSTRUCTURED NATURAL LANGUAGE DATA
  - Volume: 1.3 Billion monthly active FB users
  - Velocity: 5700 Tweets/sec. 2500 FB-msg/sec
  - Variety: scripts, languages, style, topic, …
- Today's world resides in social media
- It is impossible to process (consume, understand or summarize) this information manually.

Why NLP?

- Trending Topic Detection
- Information Retrieval & Extraction
- Information Summarization
- **Sentiment Detection**
- Rumor Detection

Challenges of NLP

- Traditionally, NLP systems are designed to handle input that is
  - Grammatically correct
  - No spelling errors
  - Single language
  - The right script
  - Text-like and Formal (unless one is working on speech interfaces)

Sentiment Analaysis

- Sentiment analysis refers to the class of computational and natural language processing-based techniques used to identify, extract or characterize subjective information, such as opinions, expressed in a given piece of text.
- The main purpose of sentiment analysis is to classify a writer's attitude towards various topics into positive, negative or neutral categories.
- Sentiment analysis has many applications in different domains including, but not limited to, business intelligence, politics, sociology, etc.
- Recent years, on the other hand, have witnessed the advent of social networking websites, microblogs, wikis and Web applications and consequently, an unprecedented growth in user-generated data is poised for sentiment mining.
- Data such as web-postings, Tweets, videos, etc., all express opinions on various topics and events, offer immense opportunities to study and analyze human opinions and sentiment.

Sentiment Analysis in Disaster Relief

- Sentiment analysis of disaster related posts in social media is one of the techniques that could gear up detecting posts for situational awareness.
- In particular, it is useful to better understand the dynamics of the network including users' feelings, panics and concerns as it is used to identify polarity of sentiments expressed by users during disaster events to improve decision making.
- It helps authorities to find answers to their questions and make better decisions regarding the event assistance without paying the cost as the traditional public surveys.
- Sentiment information could also be used to project the information regarding the devastation and recovery situation and donation requests to the crowd in better ways.
- Using the results obtained from sentiment analysis, authorities can figure out where they should look for particular information regarding the disaster such as the most affected areas, types of emergency needs.

Data Visualization/Dashboards

Information Visualization

- Information visualization focuses on providing an intuitive way of making sense of large amount of posts available in social media.
- It is widely used in social media data and contributes in many areas of exploratory data analysis, such as geographical analysis, information diffusion and business prediction.
- While most social media visualization approaches rely on geographical and temporal features, some systems exploit the semantic of the data such as sentiments to improve visualization.

Information Visualization

- What is information visualization?

  - "It is the process by which textual or numerical data are converted into meaningful images"

- Why do we need to visualize information?

  - detect the patterns hidden in text and numbers.

  - Effectively recognize the data structural features by data seekers.

- The process of recognizing patterns through human brain can facilitate users to understand the meaning of patterns more intuitively.

Data Visualization / Dashboards

- Choosing the right data visualizations is at the heart of all successful business dashboards.
- Visual analytics focuses on providing an intuitive way of making sense of large amount of big data.
- There are tons of visualizations to choose from, but not all are the right match for your data. A well-designed dashboard is compact, clear, feels familiar, and allows for rapid scanability.
- There exist some Websites that provide mashup applications to visualize and analyze tweets, including TrendsMap, Twitalyzer, and Geotwitterous.

Data visualization for disasters

- Besides disaster related data management in social media , the ability to drawing out important features is essential for better and quick understanding of situation which leads to rapid decision making in critical situations.
- Moreover, the data produced by social media during disasters and events, is staggering and hard for an individual to process. Therefore, visualization is needed for facilitating pattern discovery.
- Using visualization, people can find their answers regarding the disasters more quickly and they will figure out where they should look for to find their answers more easily.

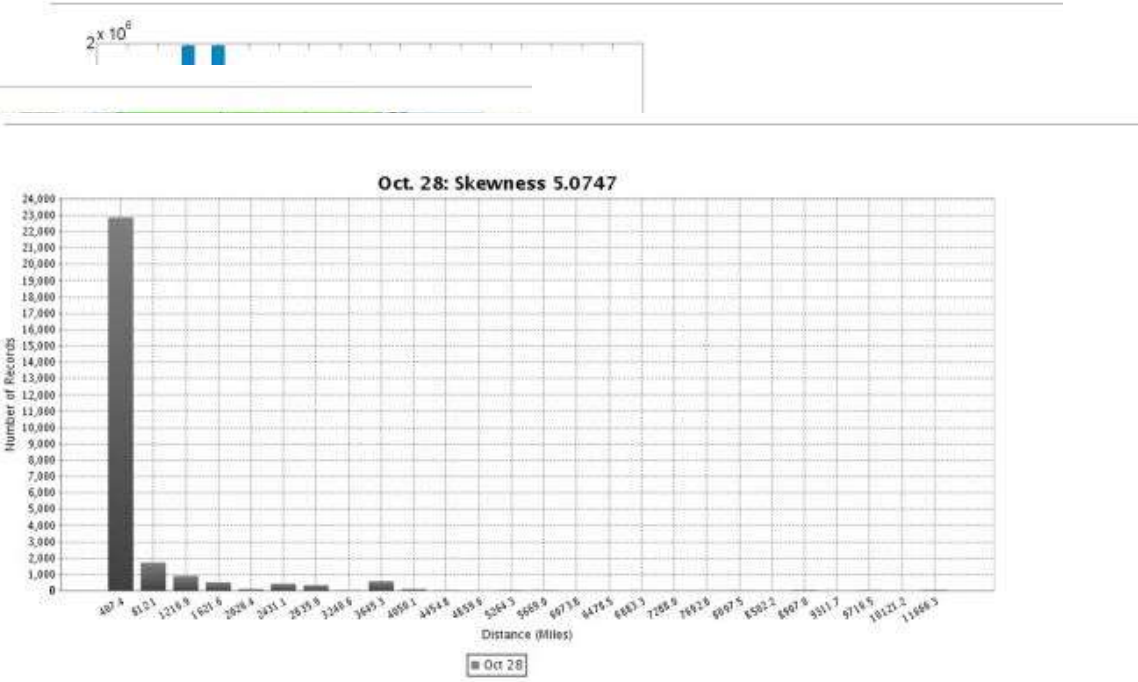
Common data visualization techniques

- **Cartogram**
- Cladogram (phylogeny)
- Concept Mapping
- **Dendrogram (classification)**
- Information visualization reference model
- **Graph drawing**
- **Heatmap**
- HyperbolicTree
- Multidimensional scaling
- Parallel coordinates
- Problem solving environment
- **Treemapping**

Common data visualizations for dashboards

1. Bar charts
2. Maps
3. Histograms
4. Line charts
5. Scatterplots
6. Sparklines
7. Pie charts
8. Gauges
9. Tables
10. Matrix View
11. Force-directed graph

Fig. 5: Hurricane Sandy impact map by FEMA [



Download high-res image (454KB) Download full-size image

Fig. 4. Histogram of October 28. The extreme positive skewness indicates short distances between each Tweet and the point where Hurricane Sandy made landfall.

Visual Analytics Challenges

- Computational complexity and scalability
- Analytic complexity
- Visual complexity
- Interaction complexity
- Evaluation complexity
- Domain/application complexity

Use of Visualization for Emergency Responders

- Visualization of data can help relief organizations accurately locate specific requests for help.
- It offers a common disaster view and helps organizations intuitively ascertain the current status.

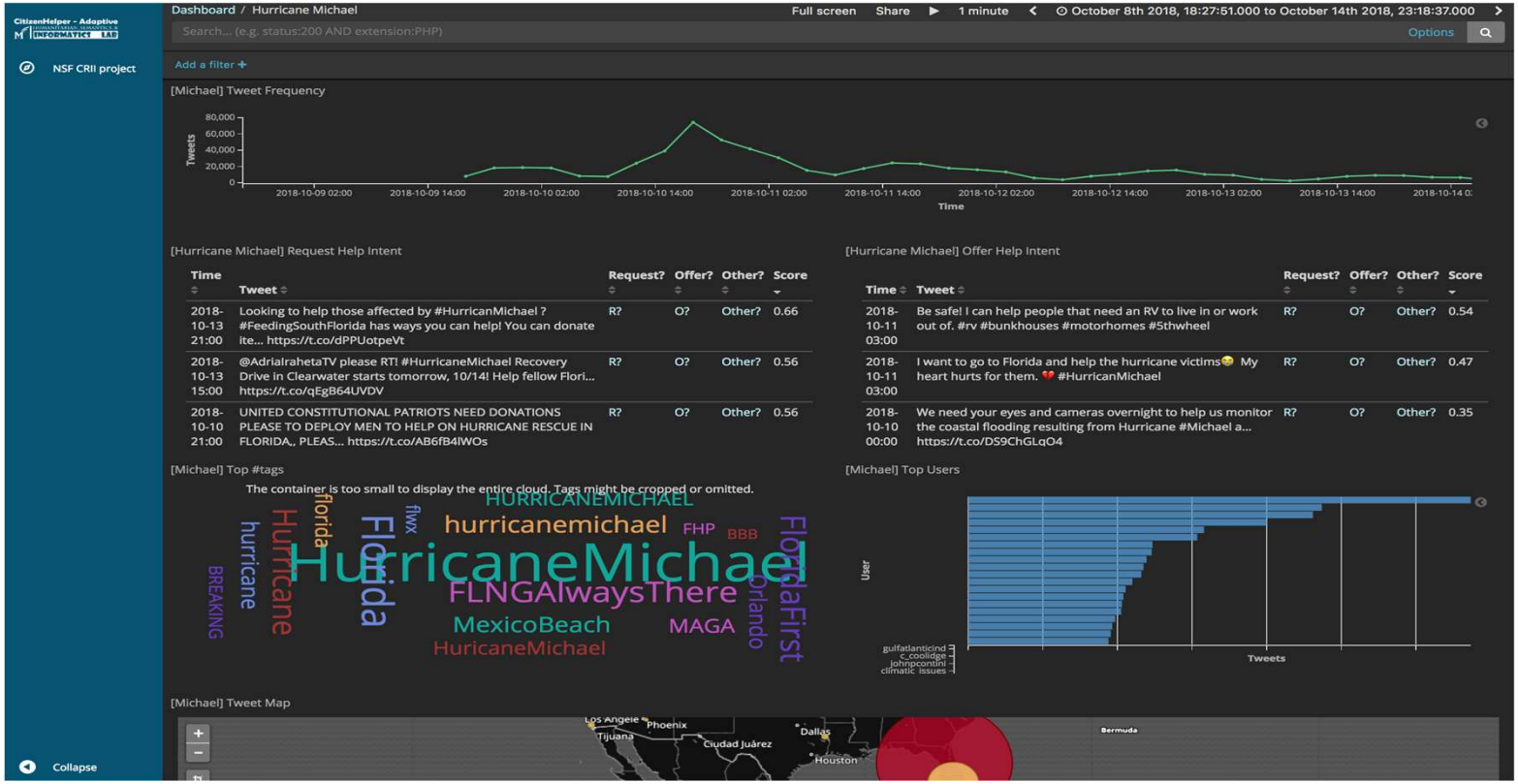
Example: data visualization

- Analysis of Twitter Data During Hurricane Sandy, 2012.

- Data set was collected for six days from Oct, 26, 2012 through Oct 31, 2012 on a slow-moving event, Hurricane Sandy.

- The map below shows Tweets per 10K people on October 28th, 2012. Tweet volume on that day was particularly low, can be see in figure.

Example 2: Visual Analytics Dashboard



Use case – Sentiment analysis from Twitter data

Sentiment analysis

- Sentiment refers to the emotion behind a social media mention online.
- Sentiment analysis is categorizing the tweets related to particular topic and performing data mining using sentiment automation analytics tools.
- We will be performing Twitter Sentiment analysis as our use case for Spark streaming.
- Sentiment analysis helps in disaster management.

Problem statement

- To design a Twitter Sentiment analysis system where we populate real time sentiments for disaster management.
- Sentiment analysis is used to:

  - Identify, extract or characterize subjective information, such as opinions, expressed in a given piece of text.

  - Classify a writer's attitude towards various topics into positive, negative or neutral categories.

  - To study and analyze human opinions and sentiment.

Use Case – Importing packages

```
//Import the necessary packages into the Spark Program
import org.apache.spark.streaming.{Seconds, StreamingContext}
import org.apache.spark.SparkContext._
import org.apache.spark.streaming.twitter._
import org.apache.spark.SparkConf
import org.apache.spark.SparkContext
import org.apache.spark.SparkContext._
import org.apache.spark._
import org.apache.spark.rdd._
import org.apache.spark.rdd.RDD
import org.apache.spark.SparkContext._
import org.apache.spark.sql
import org.apache.spark.storage.StorageLevel
import scala.io.Source
import scala.collection.mutable.HashMap
import java.io.File
```

Use case- Twitter token authorization

```
object mapr {

 def main(args: Array[String]) {
 if (args.length < 4) {
 System.err.println("Usage: TwitterPopularTags <consumer key>
<consumer secret> " +
 "<access token> <access token secret> [<filters>]")
 System.exit(1)
 }

 StreamingExamples.setStreamingLogLevels()
 //Passing our Twitter keys and tokens as arguments for authorization
 val Array(consumerKey, consumerSecret, accessToken,
accessTokenSecret) = args.take(4)
 val filters = args.takeRight(args.length - 4)
```

Use case- Dstream transformation

```
// Set the system properties so that Twitter4j library used by twitter stream
// Use them to generate OAuth credentials
System.setProperty("twitter4j.oauth.consumerKey", consumerKey)
System.setProperty("twitter4j.oauth.consumerSecret", consumerSecret)
System.setProperty("twitter4j.oauth.accessToken", accessToken)
System.setProperty("twitter4j.oauth.accessTokenSecret",
accessTokenSecret)

val sparkConf = new
SparkConf().setAppName("Sentiments").setMaster("local[2]")
val ssc = new StreamingContext(sparkConf, Seconds(5))
val stream = TwitterUtils.createStream(ssc, None, filters)

//Input DStream transformation using flatMap
val tags = stream.flatMap { status =>
status.getHashtagEntities.map(_.getText)}
```

Use case – Generating tweet data

```
//RDD transformation using sortBy and then map function
tags.countByValue()
 .foreachRDD { rdd =>
 val now = org.joda.time.DateTime.now()
 rdd
 .sortBy(_._2)
 .map(x => (x, now))
 //Saving our output at ~/twitter/ directory
 .saveAsTextFile(s"~/twitter/$now")
 }

//DStream transformation using filter and map functions
val tweets = stream.filter {t =>
 val tags = t.getText.split("
").filter(_.startsWith("#")).map(_.toLowerCase)
 tags.exists { x => true }
}
```

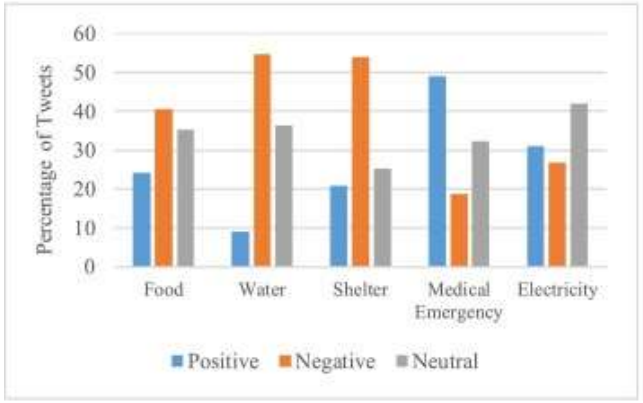
Use case – Extracting Sentiments

```
val data = tweets.map { status =>
val sentiment = SentimentAnalysisUtils.detectSentiment(status.getText)
val tagss = status.getHashtagEntities.map(_.getText.toLowerCase)
(status.getText, sentiment.toString, tagss.toString())
}

data.print()
//Saving our output at ~/ with filenames starting like twitterss
data.saveAsTextFiles("~/twitterss","20000")

ssc.start()
ssc.awaitTermination()
 }
}
```

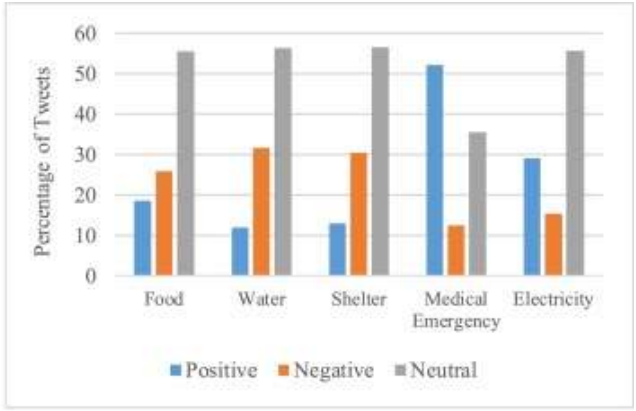
Use case - Results

Figure: Output file containing tweet and its sentiment

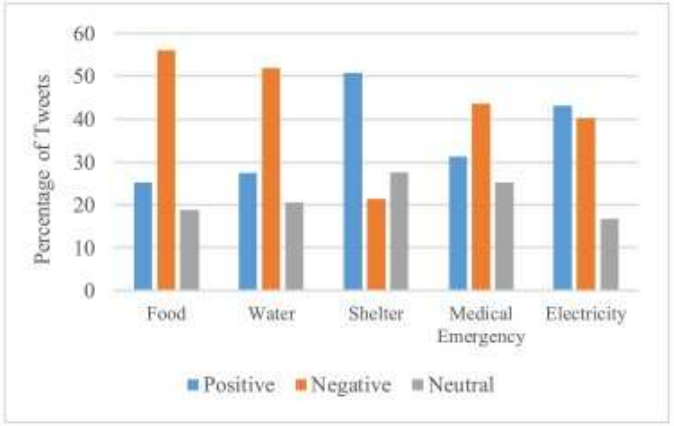
Sentiment analysis visualization



(a)



(b)



(c)

Use Case – designing a Real Time Earthquake Detection Model

Problem statement

- To design a Real Time Earthquake Detection Model to send life saving alerts, which should improve its machine learning to provide near real-time computation results.

Use Case – Requirements

1. Process data in real-time
2. Handle input from multiple sources
3. Easy to use system
4. Bulk transmission of alerts

- We will use Apache Spark which is the perfect tool for our requirements.

Use Case – Dataset

- Click here to download the complete dataset: *Earthquake Dataset – Spark Training – Edureka*.

EARTHQUAKE ROC DATASET

	S Wave							P Wave										
Classification Index	First Activity	Time Taken	Acceleration	Building Strength	Velocity	Sa	Sd	First Activity	Time Taken	Acceleration	Building Strength	Velocity	Sa	Sd	Total Weight	Sum * ROC	ROC	AVG * ROC
0	3	11	14	19	39	42	55	64	67	73	75	76	80	83	701	618.168	0.881837	623.2843
0	3	6	17	27	35	40	57	63	69	73	74	76	81	103	724	638.4503	0.881837	623.2843
0	4	6	15	21	35	40	57	63	67	73	74	77	80	83	695	612.877	0.881837	623.2843
0	5	6	15	22	36	41	47	66	67	72	74	76	80	83	690	608.4678	0.881837	623.2843
0	2	6	16	22	36	40	54	63	67	73	75	76	80	83	693	611.1133	0.881837	623.2843
0	2	6	14	20	37	41	47	64	67	73	74	76	82	83	686	604.9405	0.881837	623.2843
0	1	6	14	22	36	42	49	64	67	72	74	77	80	83	687	605.8223	0.881837	623.2843
0	1	6	17	19	39	42	53	64	67	73	74	76	80	83	694	611.9952	0.881837	623.2843
0	2	6	18	20	37	42	48	64	71	73	74	76	81	83	695	612.877	0.881837	623.2843
1	5	11	15	32	39	40	52	63	67	73	74	76	78	83	708	624.3409	0.881837	623.2843
0	5	16	30	35	41	64	67	73	74	76	80	83			644	567.9033	0.881837	623.2843
0	5	6	15	20	37	40	50	63	67	73	75	76	80	83	690	608.4678	0.881837	623.2843
0	5	7	16	29	39	40	48	63	67	73	74	76	78	83	698	615.5225	0.881837	623.2843
0	1	11	18	20	37	42	59	62	71	72	74	76	80	83	706	622.5772	0.881837	623.2843
1	5	18	19	39	40	63	67	73	74	76	80	83			637	561.7304	0.881837	623.2843

Figure: *Use Case – Earthquake Dataset*

Use Case – Flow Diagram

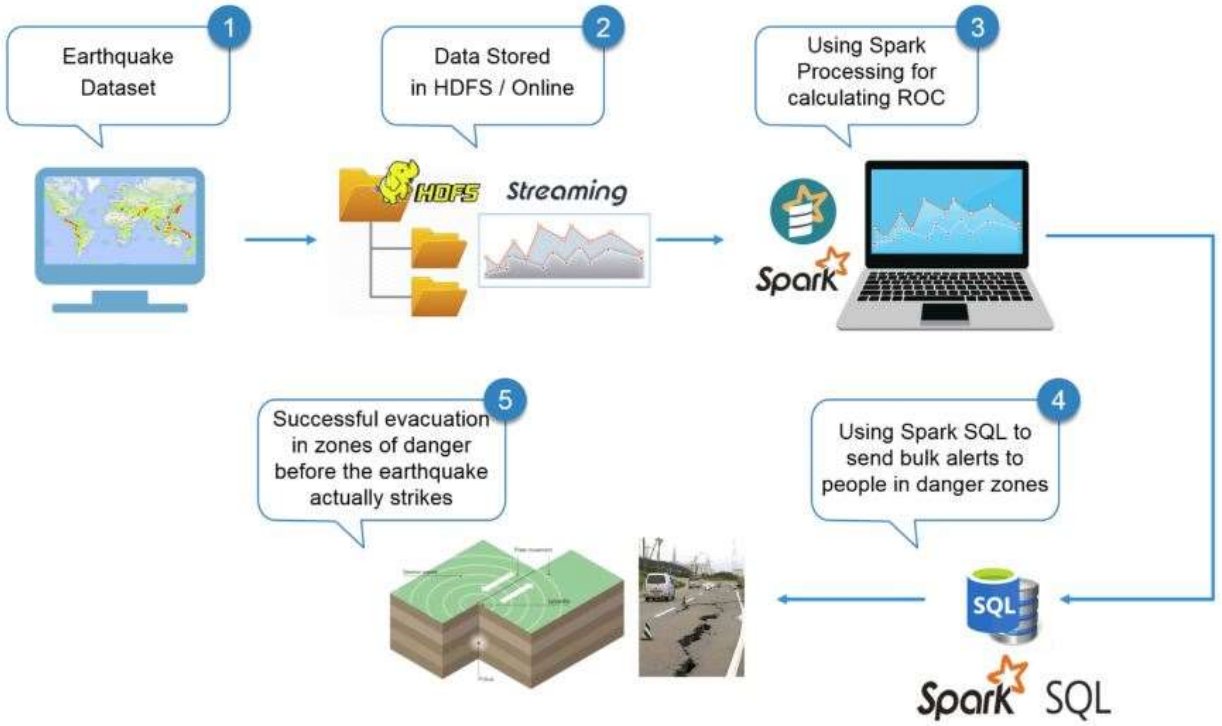


Figure: *Use Case – Flow diagram of Earthquake Detection using Apache Spark*

Use Case – Spark Implementation

- The Pseudo Code:

```
//Importing the necessary classes
import org.apache.spark._
...
//Creating an Object earthquake
object earthquake {
 def main(args: Array[String]) {

//Creating a Spark Configuration and Spark Context
val sparkConf = new SparkConf().setAppName("earthquake").setMaster("local[2]")
val sc = new SparkContext(sparkConf)

//Loading the Earthquake ROC Dataset file as a LibSVM file
val data = MLUtils.loadLibSVMFile(sc, *Path to the Earthquake File* )

//Training the data for Machine Learning
val splits = data.randomSplit( *Splitting 60% to 40%* , seed = 11L)
val training = splits(0).cache()
val test = splits(1)

//Creating a model of the trained data
val numIterations = 100
val model = *Creating SVM Model with SGD* (  *Training Data* , *Number of Iterations* )

//Using map transformation of model RDD
val scoreAndLabels = *Map the model to predict features*

//Using Binary Classification Metrics on scoreAndLabels
val metrics = * Use Binary Classification Metrics on scoreAndLabels *(scoreAndLabels)
val auROC = metrics. *Get the area under the ROC Curve*()

//Displaying the area under Receiver Operating Characteristic
println("Area under ROC = " + auROC)
 }
}
```

- Click here to [get the full source code] of Earthquake Detection using Apache Spark.

Use Case – Visualizing Results

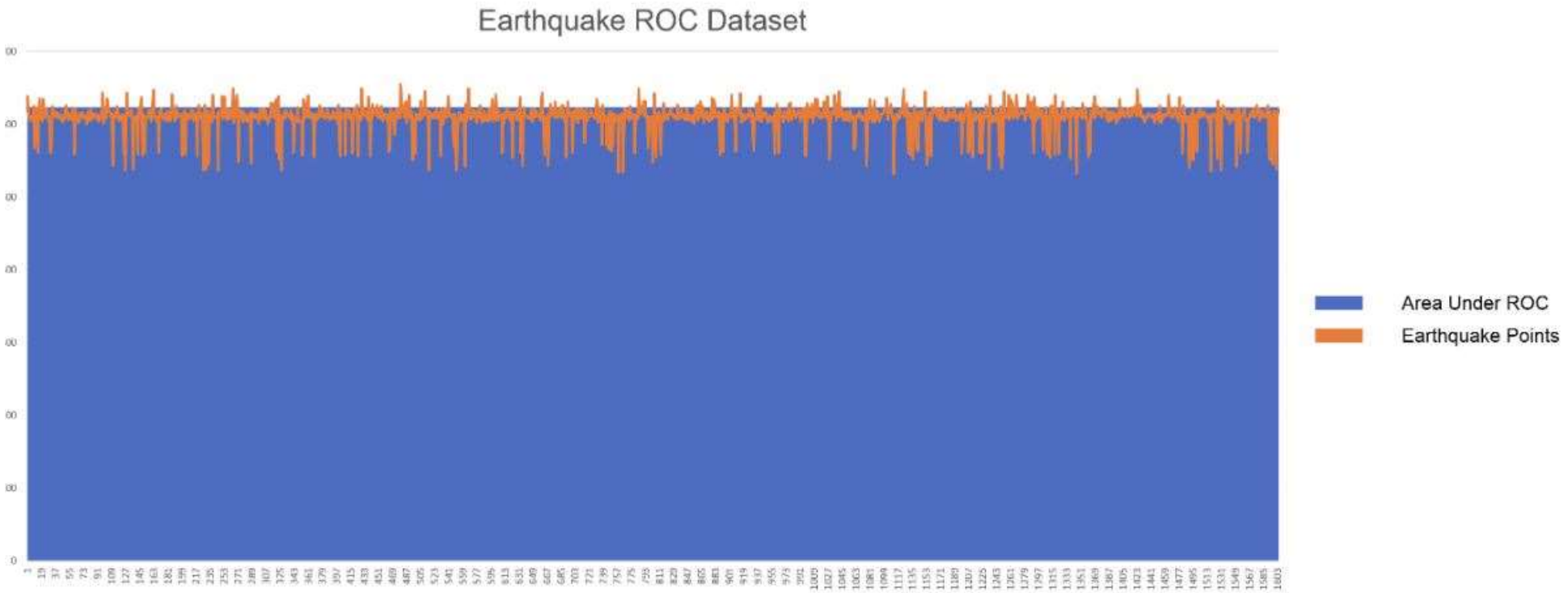


Figure: *Earthquake ROC Curve*

Use Case – Visualizing Results

Figure: *Visualizing Earthquake Points*

Survey – your evaluation on the INFO319 course

- https://docs.google.com/forms/d/e/1FAIpQLSf1OFgy7nYnaCbBHn6hoInltWv5hHhQkk_CJ5LPdqMYeQHZeA/viewform

Conclusions

- Machine learning
- Natural language processing
- Data visualization / Dashboards