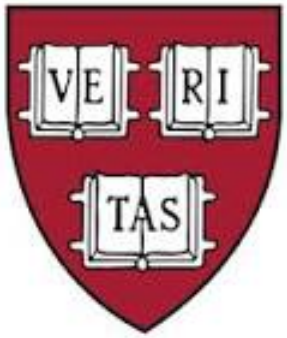



## Section 6 – Rahul Joglekar

# ML .....


CUSTOMERS WHO BOUGHT THIS ITEM:
ALSO BOUGHT:
piccolo
demotivateur.fr

# CSCI-E63, Section 6, 10-14-2017

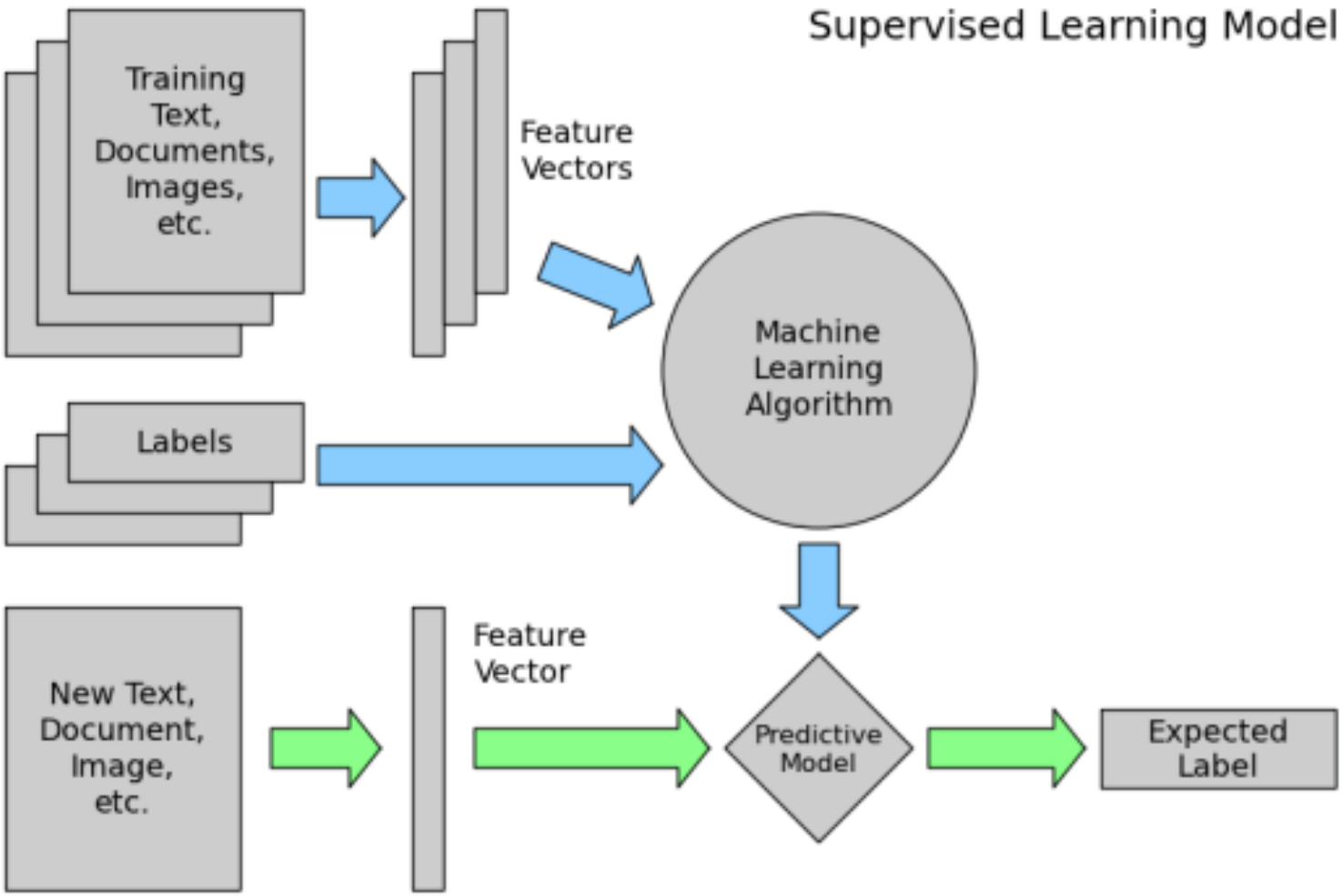
## Section Objectives

| | |
|---|---|
| 1 | ML – Logistic and Linear Regression |
| 2 | Logistic Regression Classifier - StumbledUpon |
| 3 | Linear Regression - Cars |

# Administriva – aka Usual Nagging !!

- Document your Solutions in details , please see showcased solutions
- Do NOT fall behind on the homeworks, Start the homeworks as soon as you can.

# ML Workflow


Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
Labels
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

# Spark ML and MLLib

Spark has two machine learning libraries—Spark MLlib and Spark ML—
with very different APIs, but similar algorithms. Things are influx

**MLLIB**
Original Spark ML API, based on RDDs
Spark 1.6: RDD-based APIs

**SPARK ML**
SparkML / MLPipelines / spark.ml = newer API,
Based on Dataset/Dataframe Supported API

**Which one do I choose**

- MLlib supports RDDs Vs ML supports DataFrames and Datasets
- Spark ML focuses on exposing a scikit-learn inspired pipeline API for everything from data preparation to model training.
- If you need to do streaming or online training your only option is working with the MLlib APIs – Dstream API

- For Homework you can use either APIs but we strongly recommend you use MLlib

# MLLib

- Data Types –

1. Local vector

```
import org.apache.spark.mllib.linalg.{Vector, Vectors}
>val dv: Vector = Vectors.dense(1.0, 0.0, 3.0)
>val sv1: Vector = Vectors.sparse(3, Array(0, 2), Array(1.0, 3.0))
```

2. Labeled point

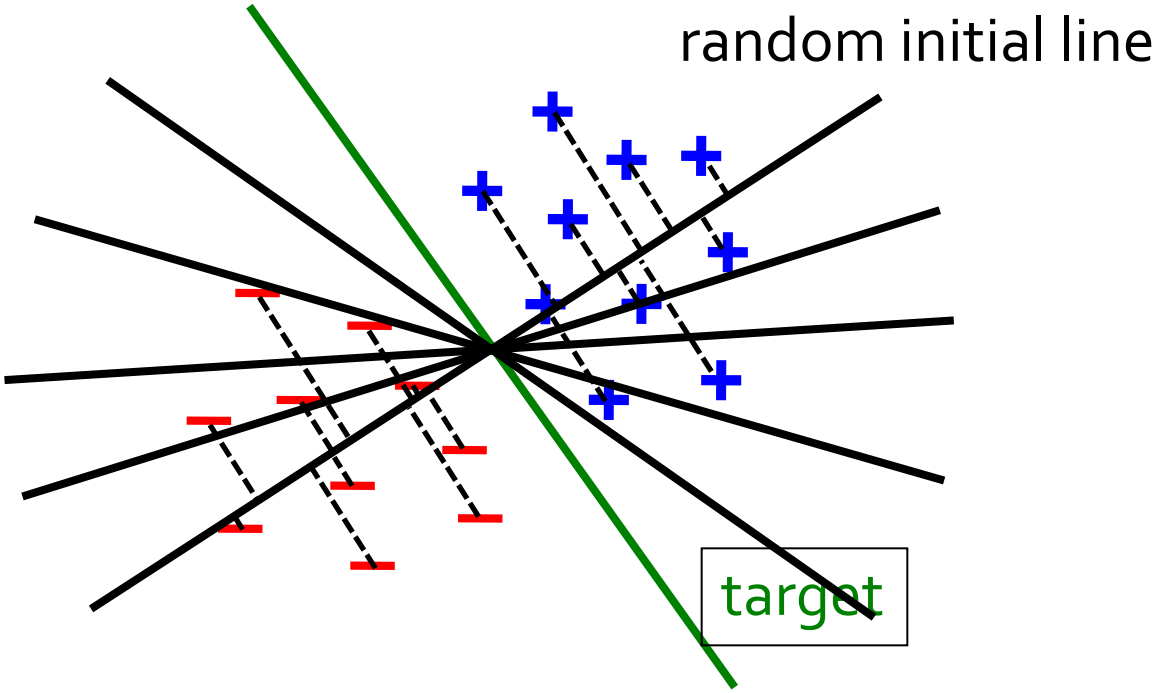
```
import org.apache.spark.mllib.linalg.Vectors
import org.apache.spark.mllib.regression.LabeledPointval
>val pos = LabeledPoint(1.0, Vectors.dense(1.0, 0.0, 3.0))
```

3. Local matrix

```
>val dm: Matrix = Matrices.dense(3, 2, Array(1.0, 3.0, 5.0, 2.0, 4.0, 6.0))
```

4. Distributed matrix
   - RowMatrix
   - IndexedRowMatrix
   - CoordinateMatrix
   - BlockMatrix

# Goal: find best line separating two sets of points



# Logistic Regression – Pseudo Code – No MLLiB

```
val data = spark.textFile(...).map(readPoint).cache()

var w = Vector.random(D)

for (i <- 1 to ITERATIONS) {
  val gradient = data.map(p =>
    (1 / (1 + exp(-p.y*(w dot p.x))) - 1) * p.y * p.x
  ).reduce(_ + _)
  w -= gradient
}

println("Final w: " + w)
```

Load data in memory once

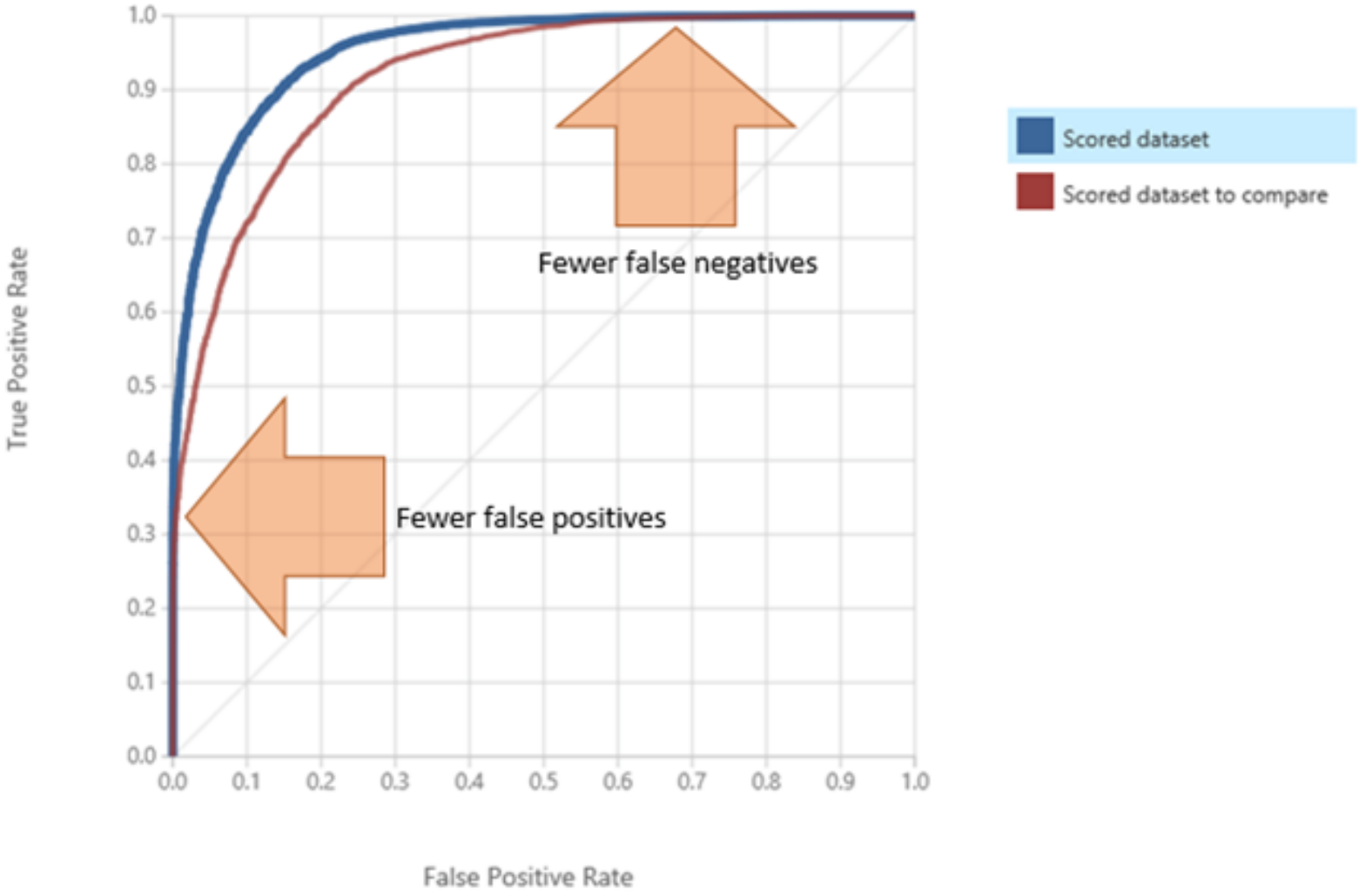
Initial parameter vector

Repeated MapReduce steps
to do gradient descent1QW

# Logistic Regression – MLLiB

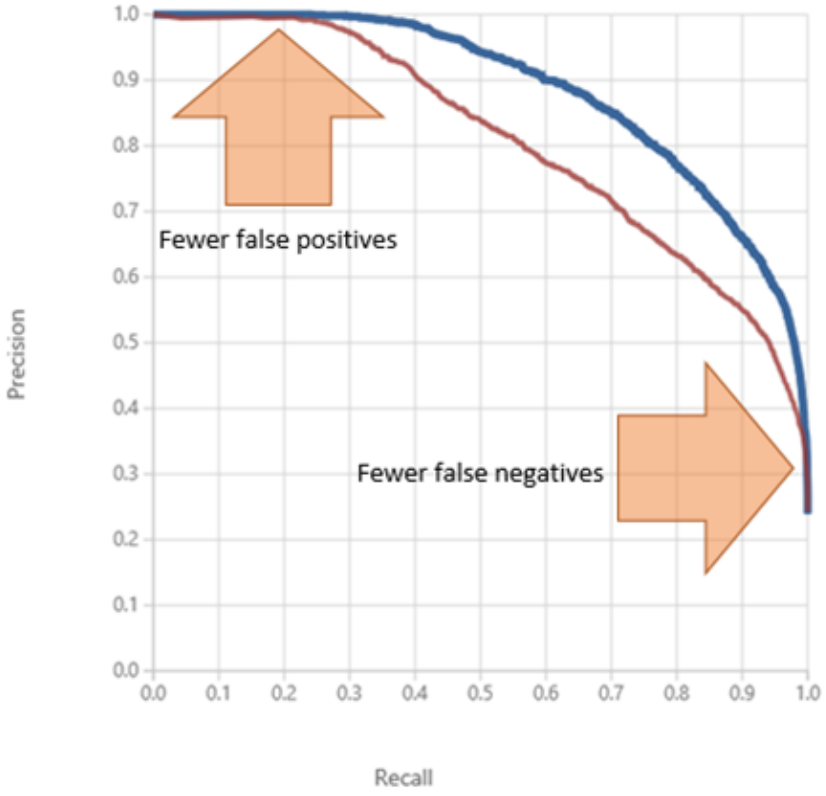
```
JavaRDD<String> spam = sc.textFile("data/spam.txt");
JavaRDD<String> ham = sc.textFile("data/ham.txt");

// Create a Logistic Regression learner which uses the LBFGS optimizer.
LogisticRegressionWithSGD lrLearner = new LogisticRegressionWithSGD();

// Run the actual learning algorithm on the training data.
LogisticRegressionModel model = lrLearner.run(trainingData.rdd());
```

# ROC




https://blogs.msdn.microsoft.com/andreasderuiter/2015/02/09/using-roc-plots-and-the-auc-measure-in-azure-ml/

# Precision and Recall



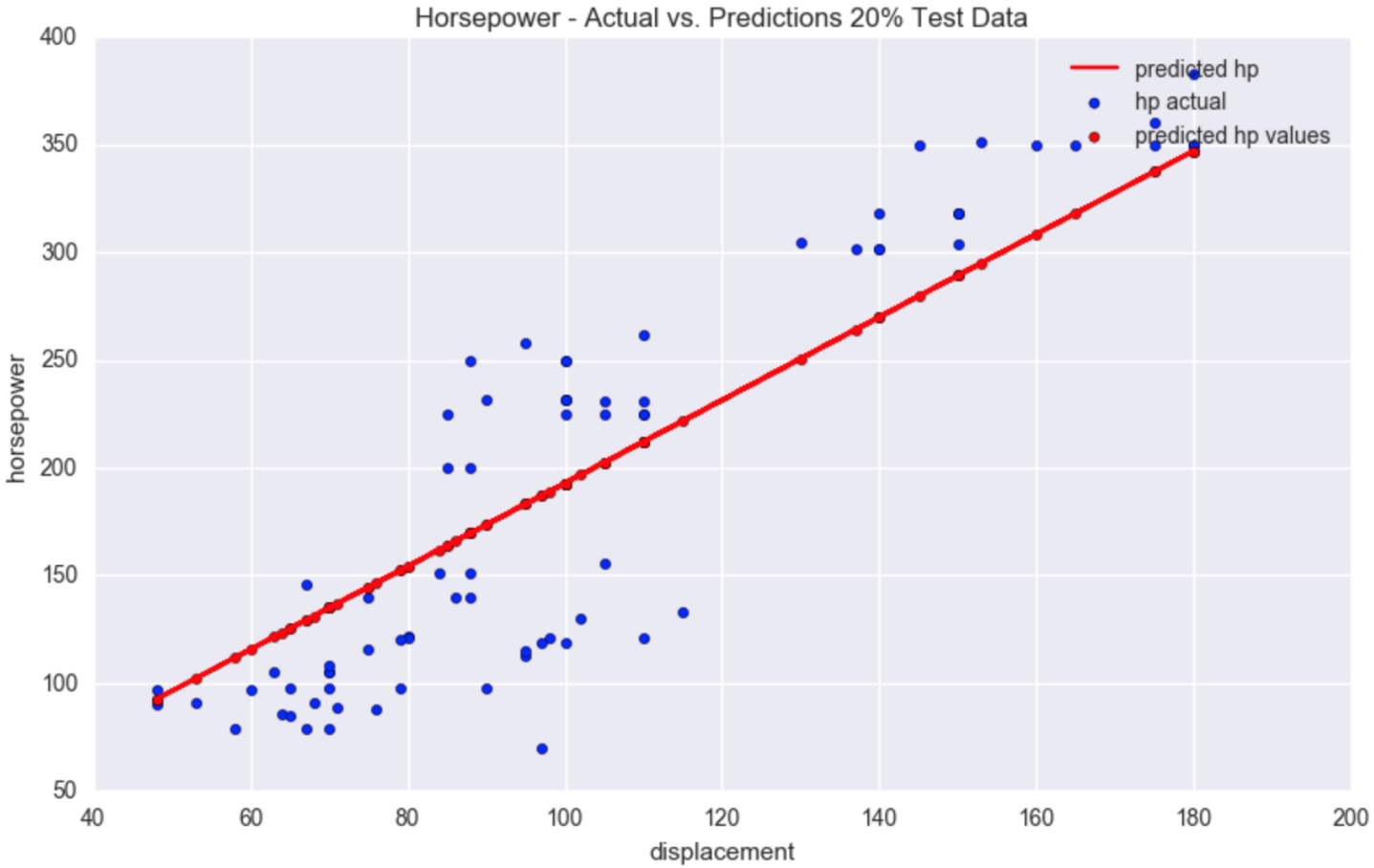

**Recall** = True Positives / (True Positives + False Negatives)
**Precision** = True Positives / (True Positives + False Positives)

https://blogs.msdn.microsoft.com/andreasderuiter/2015/02/09/using-roc-plots-and-the-auc-measure-in-azure-ml/

# Linear Regression

**Problem Statement - Build a Linear Regression Model to Predict HorsePower given Displacement**

**Refer attached Notebook**



# Logistic Regression

# Lets Write a Classifier

# SU – Kaggle Competition

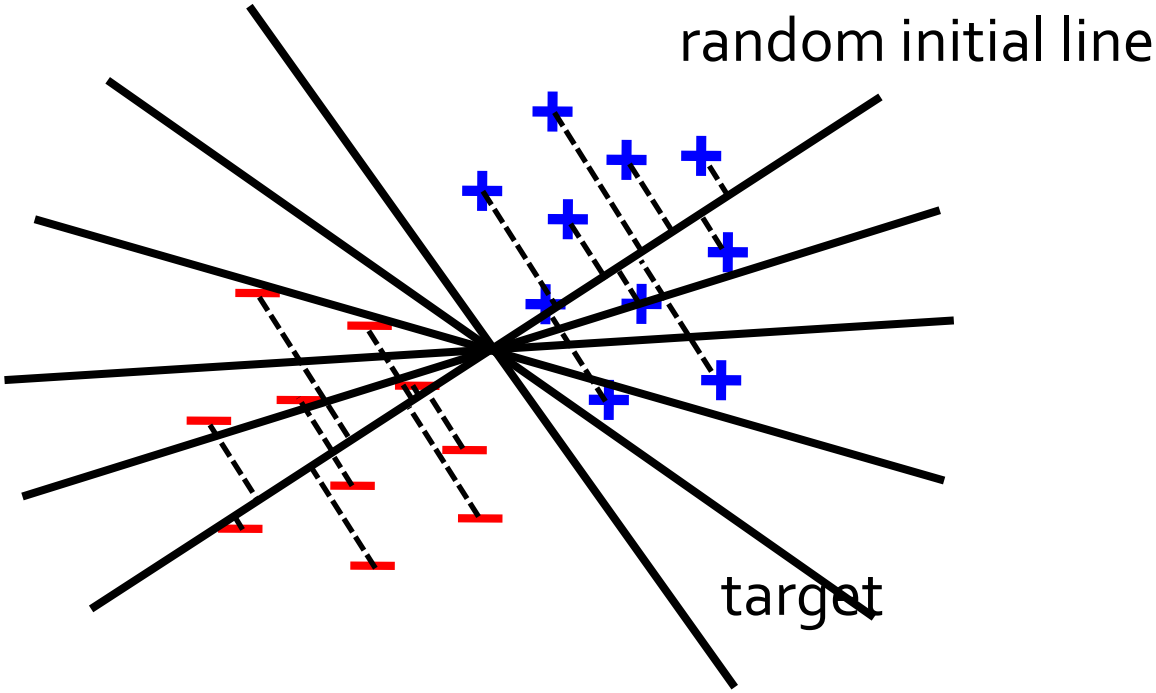
We will use the data from a Kaggle competition which classifies web pages as Ephemeral Vs Evergreen.

StumbleUpon is a user-curated web content discovery engine that recommends relevant, high quality pages and media to its users, based on their interests. While some pages we recommend, such as news articles or seasonal recipes, are only relevant for a short period of time, others maintain a timeless quality and can be recommended to users long after they are discovered.
In other words, pages can either be classified as "ephemeral" or "evergreen".

We have been given the Training and Test data along with the layout of the data , our Job is to write a Model that can do predictions going forward.

https://www.kaggle.com/c/stumbleupon

- **Goal: Write a model that will classify pages as Evergreen Vs Ephemeral OR find best line separating two sets of points**
- **Evergreen + Vs Ephemeral −**



# Data

| FieldName | Type | Description |
|---|---|---|
| url | string | Url of the webpage to be classified |
| urlid | integer | StumbleUpon's unique identifier for each url |
| boilerplate | json | Boilerplate text |
| alchemy_category | string | Alchemy category (per the publicly available Alchemy API found at www.alchemyapi.com) |
| alchemy_category_score | double | Alchemy category score (per the publicly available Alchemy API found at www.alchemyapi.com) |
| avglinksize | double | Average number of words in each link |
| commonLinkRatio_1 | double | # of links sharing at least 1 word with 1 other links / # of links |
| commonLinkRatio_2 | double | # of links sharing at least 1 word with 2 other links / # of links |
| commonLinkRatio_3 | double | # of links sharing at least 1 word with 3 other links / # of links |
| commonLinkRatio_4 | double | # of links sharing at least 1 word with 4 other links / # of links |
| compression_ratio | double | Compression achieved on this page via gzip (measure of redundancy) |
| embed_ratio | double | Count of number of <embed> usage |
| frameBased | integer (0 or 1) | A page is frame-based (1) if it has no body markup but have a frameset markup |
| frameTagRatio | double | Ratio of iframe markups over total number of markups |
| hasDomainLink | integer (0 or 1) | True (1) if it contains an <a> with an url with domain |
| html_ratio | double | Ratio of tags vs text in the page |
| image_ratio | double | Ratio of <img> tags vs text in the page |
| is_news | integer (0 or 1) | True (1) if StumbleUpon's news classifier determines that this webpage is news |
| lengthyLinkDomain | integer (0 or 1) | True (1) if at least 3 <a> 's text contains more than 30 alphanumeric characters |
| linkwordscore | double | Percentage of words on the page that are in hyperlink's text |
| news_front_page | integer (0 or 1) | True (1) if StumbleUpon's news classifier determines that this webpage is front-page news |
| non_markup_alphanum_characters | integer | Page's text's number of alphanumeric characters |
| numberOfLinks | integer | Number of <a> markups |
| numwords_in_url | double | Number of words in url |
| parametrizedLinkRatio | double | A link is parametrized if it's url contains parameters or has an attached onClick event |
| spelling_errors_ratio | double | Ratio of words not found in wiki (considered to be a spelling mistake) |
| label | integer (0 or 1) | User-determined label. Either evergreen (1) or non-evergreen (0); available for train.tsv only |

# Writing a ML Model

**First ML Model – Refer Object InitialMLModels.scala**

**Approach –**

1. Set spark context
2. Read in the training data , notice it's a tsv and not a csv , next Clean the data and create a RDD , remove headers, split on tabs , clean up quotes , convert negatives to "0"
3. Next we define the label which is col 26 , and we also start looking at features , for now we just consider numeric features col 5-25 .
4. We set various ML params like Iterations, thresholds etc etc
5. We create 5 models – **LogisticRegressionWithSGD, SVMWithSGD , NaiveBayes, DecisionTree , LogisticRegressionWithLBFGS**
6. Next we make a prediction on all our model with the same train dataset and manually calculate the accuracy for each model

# Writing a ML Model

**Improved version of ML Model – Refer Object ImprovedLR.scala**

**Approach –**

1. 50% accuracy for LR is unacceptable . What can we do now ?
2. Let us get some metrics from out of our features to better understand the data.

```
val vectors = data.map(lp => lp.features)
val matrix = new RowMatrix(vectors)
val matrixSummary = matrix.computeColumnSummaryStatistics()
```

3. We noticed the variations so clearly the data is not scaled so let us scale the data with Mllib's standard scaler

```
val scaler = new StandardScaler(withMean = true, withStd = true).fit(vectors)
```

4. Also we should not be calculating the accuracy manually lets use PR and ROC

# Writing a ML Model

**All ML Models with metrics – Refer Object MLModelMetrics.scala**

**Approach –**

1. We should look at all our models with computing the PR and ROC
2. And check their performance

spark-submit --master local[*] --class edu.hu.e63.ML.InitialMLModels ./target/scala-2.11/e63app_2.11-1.0.0.jar

spark-submit --master local[*] --class edu.hu.e63.ML.ImprovedLR ./target/scala-2.11/e63app_2.11-1.0.0.jar

spark-submit --master local[*] --class edu.hu.e63.ML.MLModelMetrics ./target/scala-2.11/e63app_2.11-1.0.0.jar

# References

https://spark.apache.org/docs/latest/

https://spark.apache.org/examples.html

Databricks - https://www.youtube.com/channel/UC3q8O3Bh2Le8Rj1-Q-_UUbA