INDUSTRIALIZATION OF ML: 01 ML Pipeline

- Suhas Hulyalkar

Aegis
SCHOOL OF BUSINESS
SCHOOL OF DATA SCIENCE
SCHOOL OF TELECOMMUNICATION

Agenda for today's discussion

- Faculty Introduction
- Introduce the module: Industrialization of ML
- Main Topic : Pipeline
- Big Picture – Data Science project workflow
    - Pipeline: relative location and significance
      Working – Indispensable: K-fold & Feature selection
      Advantages
- Case Study with Pipeline implementation
- Q & A

Industrialization of ML: 01 ML Pipeline

Purpose : Industry perspective to education

- Education perspective: Learning
- Industry : Business Solution against a Customer Contract

  : Optimize between Scope, time, cost

: Quality - fit for use,

: Automation or Systematic manual

Aegis
SCHOOL OF BUSINESS
SCHOOL OF DATA SCIENCE
SCHOOL OF TELECOMMUNICATION

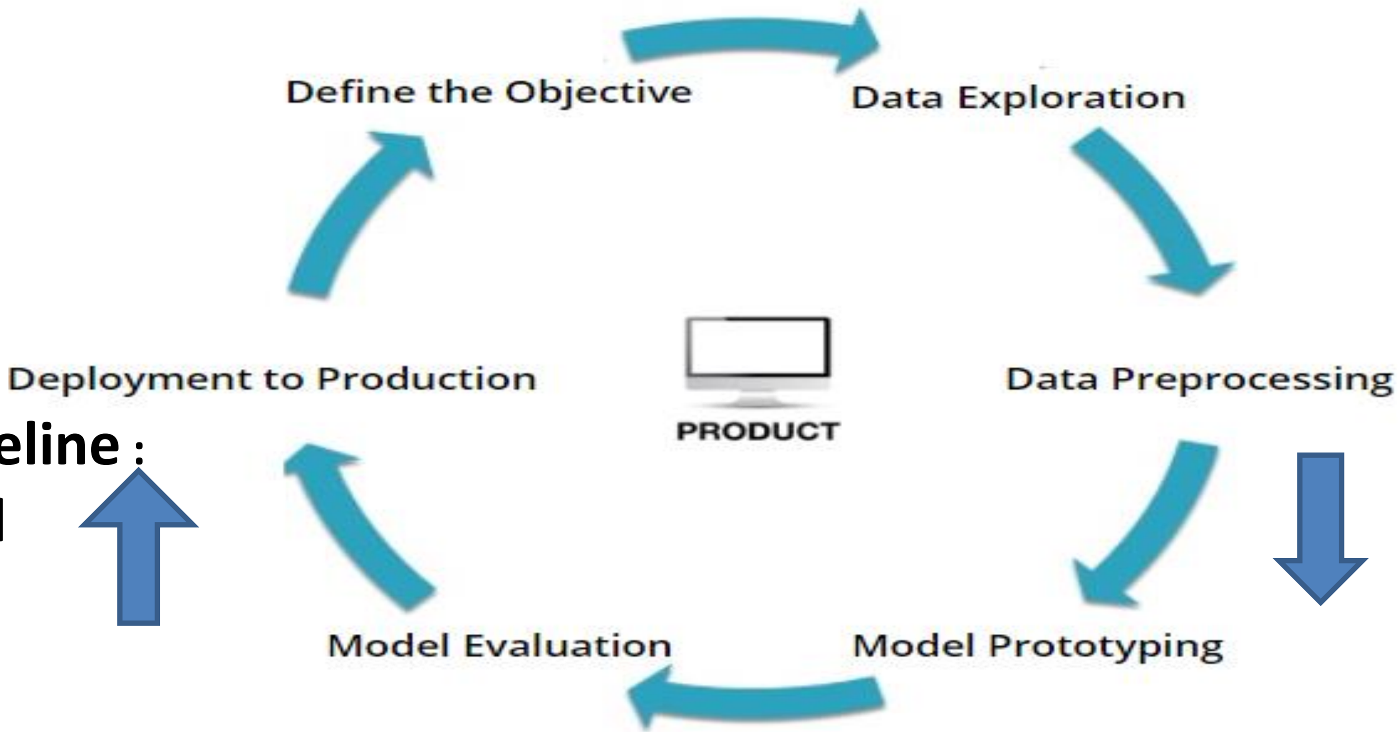
Intro: Industrialization of ML

- **Project Template -** Structure - framework; knowledge - reuse; Kick-off – Review Past projects; Plan- Estimation ; Execution – review; closure – lessons learnt
- **Club & Compare** – Multiple choices :Choose among regression, Lasso, Ridge ; Fair Comparison; common data, environment, measurement criteria
- **Pipeline** – Automation of Routine tasks
- **Save & load to Prod Library** – Serialize for saving; deserialize to load
- Brief and intuitive on theory

Pipeline: big picture

- Overall workflow of a Data Science Project

Pipeline : End



Pipeline : Start

Pipeline: big picture … contd

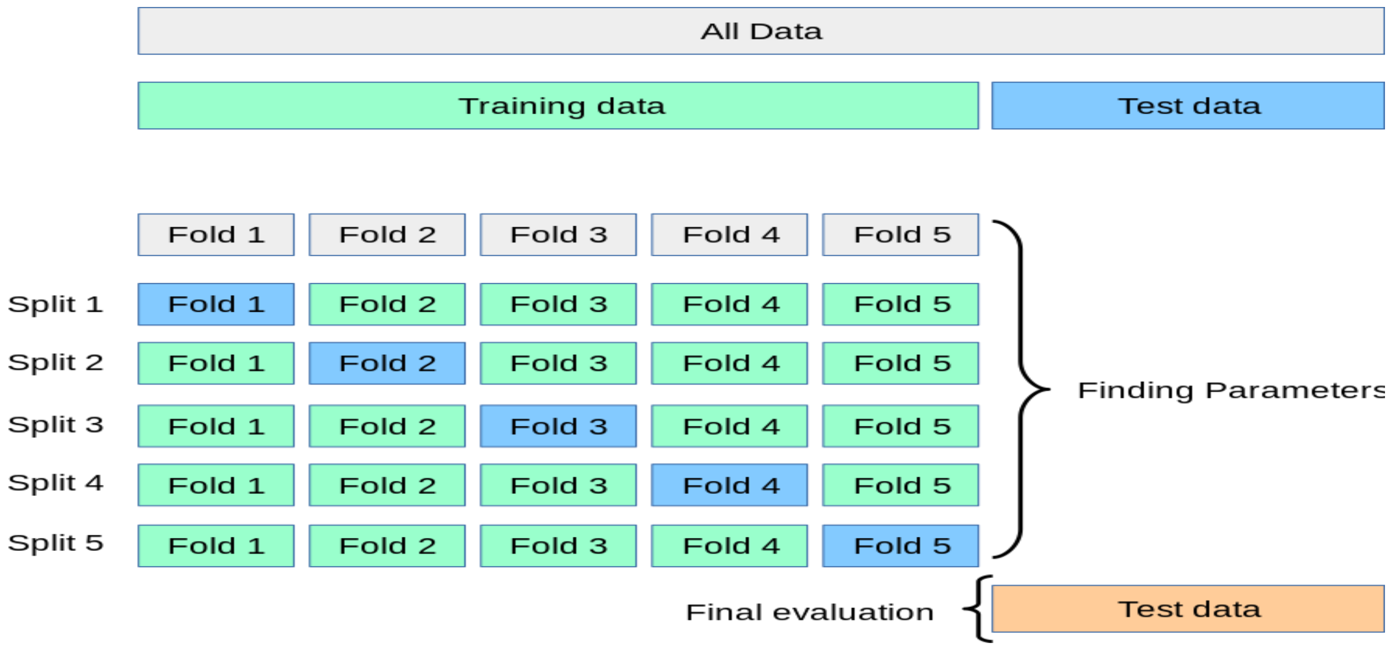
- **Preliminary Steps**
  - Data Cleaning; Imputing; Encoding; Split into train & test
  - Common factor – changes can apply uniformly; one time
  - Sklearn utilities – imputers, encoders, train-test-split
  - Preliminary steps covered in another topic : not in scope for today
- **Data Transformation & Machine Learning - Supported by Pipelines**
  - Data Transformation
    - Scaling – Standardization; Normalization; - Remove obvious biases
    - Feature Selection – PCA, RF – Retain relevant features
  - Machine Learning
    - Training - Cross Validation K-fold
    - Testing – Evaluate performance
- **Why special handling for transformers & ML**
  - Transformations twice – Training (+ CV) & testing
  - Could write a function but too rigorous

Aegis
SCHOOL OF BUSINESS
SCHOOL OF DATA SCIENCE
SCHOOL OF TELECOMMUNICATION

Pipeline … contd : Significance in Cross Val (CV)

Why special handling … contd

- Grid Search CV – tune multiple hyper parameter – tedious
- Repetitive tasks;
- Supporting to main project – ideal for automation
- Most important – risk of data leakage – shall discuss in detail



Pipeline … contd : Significance in Cross Val (CV)

- Example shows 5 fold CV
  - Training :4 folds; 5th :Test set
  - Reject model; retain result
  - Repeat each split
  - Performance : avg value
- Before ML say, we need to standardize –
  - Should we standardize entire training set?

    If yes -

    - For standardization : scaling factors (X- **μ**) /**σ**
    - At training: the system already knew entire distribution
    - knowledge was stamped into the rescaled values
- Pipelines helps us prevent the data leakage. How?
  - It ensures that transformation is constrained to each fold of CV

Working of Pipeline Class

- Allows chaining of multiple processes into a single Estimator
- Estimator : An object that learns from data
  - Imagine ML algorithms & transformer processes defined as a class
  - Estimator as an instance of
    - Algorithm class – Classifier, Regressor
    - Transformer class – Standardizer, Normalizer
  - Algorithms learn parameters like slope, Y-intercept (y=**a0**+**a1**X+…)
  - Transformers learn parameters like
    - Scaler(**min, max**);
    - Standardizer (μ(**mean**), σ(**stdev**))

      And apply to the feature to transform or rescale the data
- Estimator object encapsulates complete logic for each routine step into a function call
- We select and define a sequence of function calls we want

Working of Pipeline Class … contd

- Function calls are coded in the form of a list of tuples with
  - Name & instance of transformer or algorithm

```
# Create pipeline
estimators = []    #Define a list
estimators.append(('standardize', StandardScaler()))
estimators.append(('lda',
LinearDiscriminantAnalysis()))
```

- All except the last, estimators in a list should have a transform method
- Last estimator can be a transformer or algorithm – decide the output
- When we run the code,
  - function calls perform the specified individual functions,
  - fit them to the training data &
  - apply it to the test data

  without we having to code the repetitive details

Working of Pipeline Class … contd

- Advantages of Pipelines
  - Allows enforcing desired order of application steps
  - Ensures Reproducibility of results
  - Creates a convenient work-flow – easy to follow, implement
  - Avoid data leakages from your test data into the trained model in cross- validation

Case Study : Implement a Pipeline:

Example 1: **Set up a pipeline with - Standardization and LDA**

Data Overview & Problem Statement

- Our dataset contains the medical records of patients being assessed for diabetes.
- Called Pima Indians Dataset by UCI ML Repository.
- Pima Indians are the Native Americans from Arizona state of USA. They were found prone to obesity and diabetes.
- (Next slide lists various features i.e. the diagnostic measurements
- The last col 'class' is the diagnosis whether patient has diabetes (1) or not (0)

- The objective is to predict whether a given new patient is likely to get diabetes in the next 5 years based on the measurements

Pima Indians Dataset : Fields description

Input Features

- **preg** = Number of times pregnant
- **plas** = Blood Glucose : 2-hr-val 140 - 200 prone ; > 200 : diabetic
- **pres** = diastolic BP up till 60 yrs then down,
- **skin** = Triceps skin fold thickness (mm) : activates own insulin. helps control **test** = 2-Hour serum insulin (mu U/ml)
- **mass** = BMI (wt. kg/ht m^2); BMI above normal : risk of diabetes
- **pedi** = Diabetes pedigree function
- **age** = Patient's Age (years)

Target variable

- **class** = Class variable (1: +ve, 0: -ve for diabetes) : binary (0/1)

Pima Indians Dataset : values & metadata

```
No. of Rows 768, Columns 9
class
0    500
1    268
```

	preg	plas	pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0
8	2	197	70	45	543	30.5	0.158	53	1
9	8	125	96	0	0	0.0	0.232	54	1

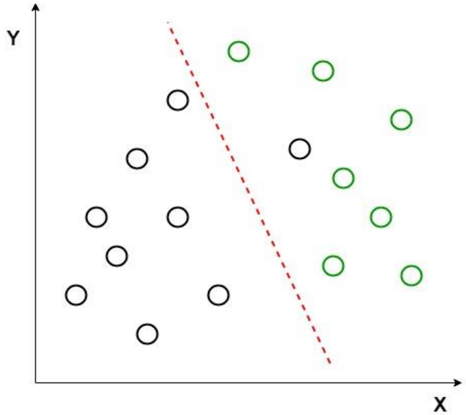
Pima Indians Dataset : Correlation Coeff

	preg	plas	pres	skin	test	mass	pedi \
preg	1.000000	0.129459	0.141282	-0.081672	-0.073535	0.017683	-0.033523
plas	0.129459	1.000000	0.152590	0.057328	0.331357	0.221071	0.137337
pres	0.141282	0.152590	1.000000	0.207371	0.088933	0.281805	0.041265
skin	-0.081672	0.057328	0.207371	1.000000	0.436783	0.392573	0.183928
test	-0.073535	0.331357	0.088933	0.436783	1.000000	0.197859	0.185071
mass	0.017683	0.221071	0.281805	0.392573	0.197859	1.000000	0.140647
pedi	-0.033523	0.137337	0.041265	0.183928	0.185071	0.140647	1.000000
age	0.544341	0.263514	0.239528	-0.113970	-0.042163	0.036242	0.033561
class	0.221898	0.466581	0.065068	0.074752	0.130548	0.292695	0.173844

	age	class
preg	0.544341	0.221898
plas	0.263514	0.466581
pres	0.239528	0.065068
skin	-0.113970	0.074752
test	-0.042163	0.130548
mass	0.036242	0.292695
pedi	0.033561	0.173844
age	1.000000	0.238356
class	0.238356	1.000000

Implement a Pipeline: Plan

- To create a pipeline object, we need to provide a list of steps.
- In the first example our steps are -
  - Standard Scaler (transformer) &
  - Linear Discriminant Analysis (algorithm)
- Why LDA? : Part of another session 'Club & Compare.' For now - go with this plan



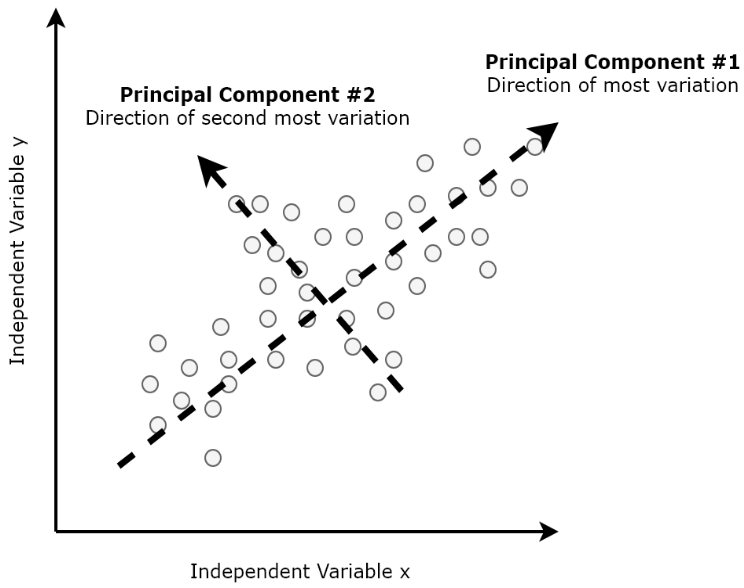
- In LDA - we find a line (or plane or hyperplane) that will maximize distance between the means and minimize the scatter of each group

Implement a Pipeline: Plan

- Divide the data-set into training and test-set with a seed
- We shall do an 80/20 split - helper function train_test_split
- List of tuples - name and an instance of transformer or estimator
- With the function call we basically instantiate the estimator object within the pipeline and the system internally allocates space to support 10 folds CV
- Model will be evaluated using helper function cross_val_score.
- After checking out the CV accuracy, we shall go forward with final evaluation of the model performance using the test data
- Accuracy will be measured in terms of %age data correctly classified out of total samples

Implement a Pipeline: Feature Union Class

- In the extended example we have –
  - Step1 : Feature Union (SelectKbest & PCA) similar to pipeline
  - Step 2: Logistic regression (algorithm) & Feature union as estimators



PCA – Dimensionality reduction technique that preserves the most important features with maximum variation of the data. PCA reduces the dimension in such a way that new variables are orthogonal to each other (i.e. they are independent or not correlated)

Implement a Pipeline: Feature Union Class … Contd

SelectKBest – if the features are quantitative- as in our example, compute the ANOVA F-value between each feature and the target vector. The F-value scores examine if, when we classify the numerical feature by the target classes (our example 0 or 1), the means for each group are significantly different

- This example shows how to use Feature Union to combine features obtained by PCA and SelectKbest algorithm.
- It will create a union of features identified by both methods
- The Pipeline step will be followed by 10-Fold CV like we did in 1st Example
- Rest of the process will be identical

To Jupyter Notes …

SCHOOL OF BUSINESS
SCHOOL OF DATA SCIENCE
SCHOOL OF TELECOMMUNICATION