Week 1 Introduction to Data Mining and Machine Learning

Theory and Practice

Ying Lin, Ph.D

Jan 17, 2020

Definition of Data Mining

- **Non-trivial extraction** of **implicit**, **previously unknown** and **potentially useful** information from data (by Gregory Piatetsky-Shapiro)
- Origins
  - Machine learning
  - Statistics
  - Database system

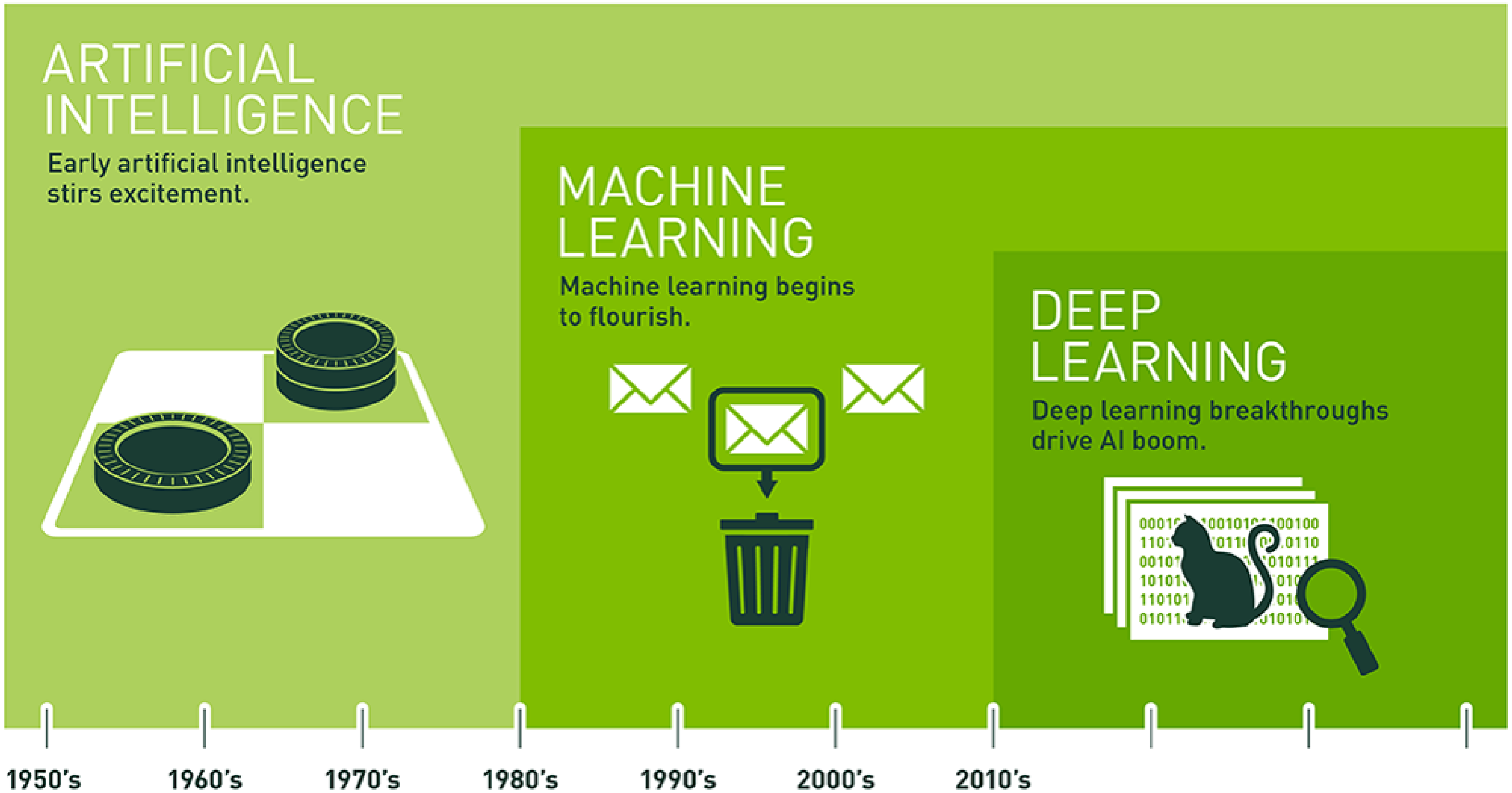
Machine Learning Framework

- Algorithms that learn from the data/past experiences, instead of being instructed step-by-step how to solve a problem
  - Training involves fine turning the algorithm (hyper-parameters tuning) which is key to identify the best model (ML algorithm parameters)
  - Hyperparameter tuning methods: grid vs. random search etc.
  - Example: for deep learning algorithm, hyper-parameters include activiation function, objective function, number of hidden layers, number of nodes in each layer, learning rate, epoch, size of mini-batches, drop-out rate, etc.
- Difference between machine learning and statistics
  - Hypothese search space vs. mathematical modeling
  - Non-parametric vs. often parametric (assumptions)
  - Hypothese-free vs. hypothese-driven
  - Training vs. curve-fitting

Common Themes in Machine Learning

- Common issues in learning
  - Overfitting
  - Bias-variance decomposition
  - Hyper-parameter optimization
- Trial-and-Error Approach
  - Hypothesis space searching
  - Guided by loss function (Error)
  - Optimization algorithms: e.g. gradient descent

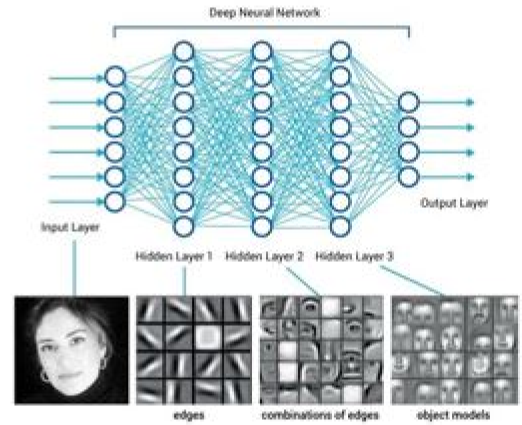
Evolution of AI: Timeline



Breakthrough of AI

Data

Computational Power

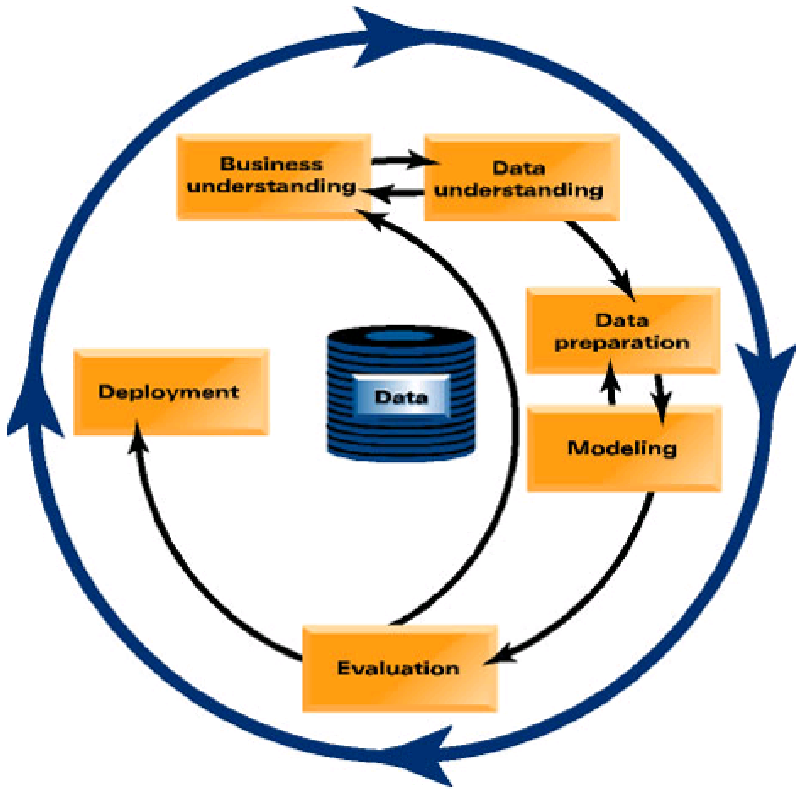


Deep Learning Algorithms

6/24

Data Mining Process: CRISP-DM Overview

- CRISP-DM (CRoss-Industry Standard Process of Data Mining): an iterative process



Predictive vs. Descriptive Analytics

- Descriptive analytics: derive patterns (correlations, trends, clusters, and anomalies) that summarize the underlying relationship in the data
  - Data aggregation, summary/descriptive statistics and unsupervised learning algorithms
  - Insights into the past, answering “What has happened”
- Predictive analytics: construct model from past data to predict unknown or future outcome of events
  - Inferential statistics and various supervised/unsupervised machine learning algorithms
  - Insights into the future, answering “What will happen”
- Prescriptive Analytics: an extension of predictive analytics
  - Guided by insights from predictive analytics, we could simulate different outcomes to identify decisions that could optimize the outcome
  - Focus on proactive decisions/actions and require substantial domain knowledge

Different Ways to Classify Machine Learning Algorithms

- Supervised vs. un-supervised vs. semi-supervised
  - Availability of class labels
  - Reinforement learning: maximize reward
- Transparent vs. blackbox
  - Interpretability of the models
- Eager vs. lazy
  - Presence of explicit model induction

Data Mining Task 1: Classification

- Goal: map data points into a discrete set of categories
- Real world examples
  - Spam email detection
  - Credit card application approval
  - Text classification
  - Fraud detection

Breakdown of a Classification Problem

- Goal
- Example
- Target classes/categories
- Source of training data
- Attributes

Data Mining Task 2: Clustering

- Goal: identify underlying grouping structure of data
- Applications
  - Customer segmentation
  - Biomedical: microarray analysis
  - Data exploration

Data Mining Task 3: Association Rule Mining

- Goal: find items that co-occur frequently among a set of transactions and output association rules
- Applications
    - market basket analysis
    - medical diagnosis
    - Recommender system

Data Mining Task 4: Regression

- Goal: model continuous variable (linear regression, decision tree regression, Support Vector Regression (SVR), random forest regression) or categorical variable (logistic or probit regression)
- Applications
  - Stock price prediction
  - Classification applications

Data Mining Task 5: Anomaly Detection

- Goal: detect significant deviations from normal behavior. Could be considered as a special case of classification.
- Applications
  - network inrusion detection
  - credit card fraud detection

R vs. Python Comparison

- De Factor programming languages for data science
  - open-source programming language with large and active developer communities
  - Extensive data analyis and machine learning libraries and APIs
- R
  - A more specialized programmining language and excels in statistical analysis, data visualization, data reporting and presentation
  - User base: scholars and researchers in academia
- Python
  - A general-purpose language and excels in deep learning, model deployment, web programming, integration with other systems
  - User base: programmers and developers in industry

Data Wrangling and Munging in R

- *dplyr* package
  - key verbs for data manipulations: group_by(), summarise(), arrange(), filter(), select(), mutate()
  - Other verbs: join(), distinct(), rename()
- *tidyr* package
  - Tidy format (vs. messy format): a row is a record and each attribute/feature is a column, populated by Hadley Wickham
  - gather(): data transpose/reshaping from wide to tall format
  - spread(): from tall to wide format
- pipe operator (%>%)
  - Chain together multiple data manipuation steps to avoid nested function calls

Machine Learning Package in R: *caret*

- **CARET** (Classification And Regression Training): one-stop solution for data analytics
- Provide wrapper functions to 200+ machine learning algorithms
- Standardize the function names, syntax and parameters
  - common ML functions: train(), predict(), etc.
  - Common function arguments: method=, metric=, tuneLength=, tuneGrid= , control=, trControl=, etc.
- Provide functions that cover the complete data mining process
  - Data preprocess and prepare: preProcess(), createDataPartition(), dummyVars()
  - Data modeling: train(), predict()
  - Model parameter tuning (hyperparameters and model-specific parameters): trainControl(), expand.grid()
  - Model performance and attribute evaluation: confusionMatrix(), defaultSummary(), varImp()
  - Model saving and deploying: saveRDS(), readRDS()

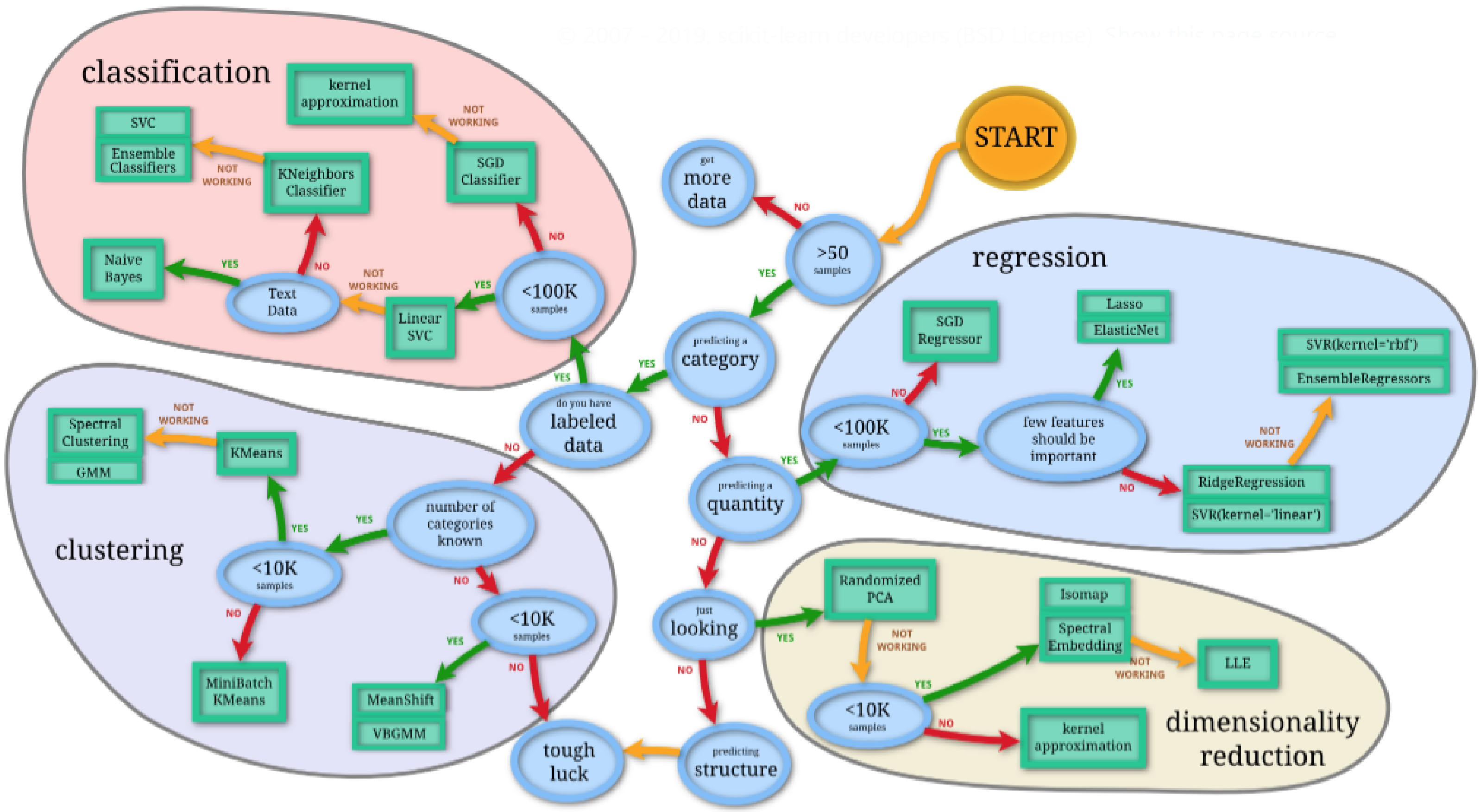
Machine Learning Package in Python: *sklearn*

- Preprocessing
  - LabelBinaizer(), LabelEncoder(), OneHotEncoder(), StandardScaler(), scale()
  - fit_transform(), transform()
- Modeling workflow
  - Common interface: fit(), predict(), predict_proba(), score()
  - model_selection: train_test_split(), GridSearchCV()
  - pipeline: make_pipeline(), Pipeline()
  - Model tuning: set_params(), get_params(), GridSearchCV()
- Model performance evaluation
  - metric: confusion_matrix(), accuracy_score(), mean_squared_error(), roc_curve(), classification_report(), r2_score()
  - cross_validation: StratifiedShuffleSplit()

sklean Continued

- Machine Learning models
  - SVM: SVC(), linearSVR(),
  - ensemble: RandomForestRegression(), AdaBoostClassifier(), GradientBoostingClassifier()
  - decomposition: PCA(),
  - linear_model: lasso(), ridge(), LogisticRegression(),
  - naive_bayes: GaussianNB(), MultinomialNB(), BernoulliNB()
  - Tree: DecisionTreeClassifier(),
  - neighbors: KNeighborsClassifier(),
  - discriminant_analysis: LinearDiscriminantAnalysis(), QuadraticDiscriminantAnalysis()
- model deployment
  - externals.joblib: dump(), load() models in ".pkl" format

Overview of *sklearn*



21/24

R Authoring Framework: *rmarkdown*

- Three components
  - YAML metadata: block enclosed in - - - and follow YAML syntax (key-value pair)
  - Executable R Code chunks: block of ` ` `{r} … ` ` `, work with knitr
  - Text description: follow markdown language syntax
- Could be use to produce report, presentation slides, website, etc. to integrate codes, outputs and analyis
- Reference
  - Knit the R code chunk output rmarkdown::knitr
  - Output the knit and markdown results to one of three formats: PDF, WORD, HTML
  - Render the rmarkdown file include both knit and output
  - R Markdown Intro

22/24

Interactive Document and Web Framework: *shiny*

- Make interactive and real-time data analytics accessible to the public
- Reactive programming: make the outputs react to the user specified inputs
- Integrate HTML/CSS/JavaScript into the web app without prior knowledge
- Server/UI design: one-file system
  - Server: include logic of the web app and engine of data analytics
  - UI (User Interface): create layout of the app and use Shiny functions to generate HTML
- Demo of *shiny*

Expectations for the Course: After the Live Session

- Reproduce all the codes provided in the slides for each live session, check the outputs for yourself
- Make sure you are familiar with all the packages and functions used in the lecture notes
- Adapt, revise, and expand the codes to explore questions that interest you using the same demo datasets
- Demo datasets: employee churn, income, car, crime, email spam, medical, housing price, text, etc.
- Spend substantial amount of time coding with toy datasets and take advantage of the best teacher ever … **GOOGLE**