End to End Framework for Developing Machine Learning Solution

Presenter: Karan Vijay Singh

UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS

Introduction to Machine Learning

- It is a discipline where computer programs make predictions or draw insights based on patterns they identify in data and are able to improve those insights with experience — without humans explicitly telling them how to do so.

- At a conceptual level, we're building a machine that given a certain set of inputs will produce a certain desired output by finding patterns in data and learning from it.

- For example, given the area in square feet,furnished, address, parking and number of bedrooms (the input) we're looking to predict a home's sale price (the output).

Why designing a Machine Learning Solution is different from designing a Software solution?

For software product,

- Testing a software system is relatively straight-forward as compared to testing an ML solution as you know the desired outcomes in software.
- Also in Traditional software engineering, using encapsulation and modular design helps us to create maintainable code and it is easy to make isolated changes and improvements.

But in ML,

- Desired outcome depends on kind of problem you are trying to solve.
- You may get the output you are looking for but it's not always the case.
- Desired behavior cannot be effectively implemented in software logic without dependency on external data.

Motivation

- Google Scholar Search -> No papers on RE for ML but lot on ML for RE.
- No authoritative source available that can be consulted when designing a machine learning solution.
- This is an attempt to develop an end to end framework for developing machine learning solutions.

Machine Learning Problem Framework

Presenters:
Bikramjeet Singh(20752928)
Priyansh Narang (20716980)

RE for ML/ ML for RE feat. Google Scholar

- We tried multiple keywords to review past work done in this space: requirement engineering, requirement elicitation, SDLC for ML
- No credible source for RE for ML
- Several papers where authors have used techniques from ML to improve Requirement Engineering:
  - Estimation of effort for tasks
  - Prioritizing requirements
- Few online publishing platforms have articles about the intersection of SE and ML
- This is an attempt at developing and end-to-end framework for systems leveraging Machine Learning

Formal definition

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E”

Type of Machine Learning Problems

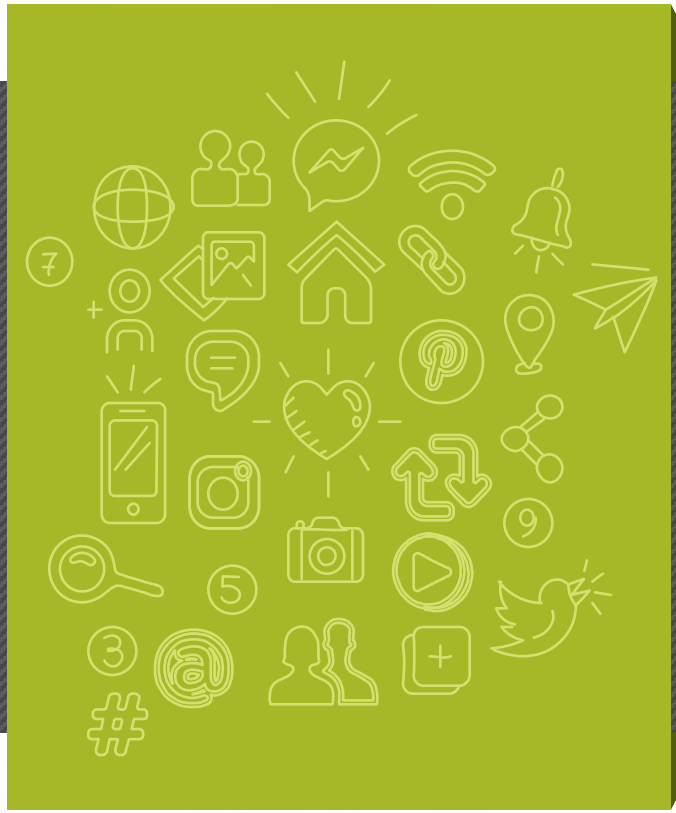
Type of ML Problem	Description	Example
Classification	Pick one of N labels	cat, dog, horse, or bear
Regression	Predict numerical values	click-through rate
Clustering	Group similar examples	most relevant documents (unsupervised)
Association rule learning	Infer likely association patterns in data	If you buy hamburger buns, you're likely to buy hamburgers (unsupervised)
Structured output	Create complex output	natural language parse trees, image recognition bounding boxes

3. What are your success metrics?

- How do you know the system has succeeded? Failed?
- Phrased independently of evaluation metrics
- Tied to the ideal outcome
- Domain/product/team specific
- Are the metrics measurable?
- When are you able to measure them?
- How long will it take for you to know that system is a success or failure?
- Example: Predict the credit limit within 10% range of the manual process
- Example: Reduce the time taken to approve the user for a certain credit limit by 90%

4. What’s the ideal output?

- Write the output you want your models to produce in plain english
- The output must be quantifiable that the machine is capable of producing
- For instance: “User did not enjoy the article” produces much worse results than “User down-voted the article”
- For your ideal output, can you obtain example outputs for training data?



RE for Data Cleaning with Machine Learning
CS 846

Presenter: Ishank Jain

UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS

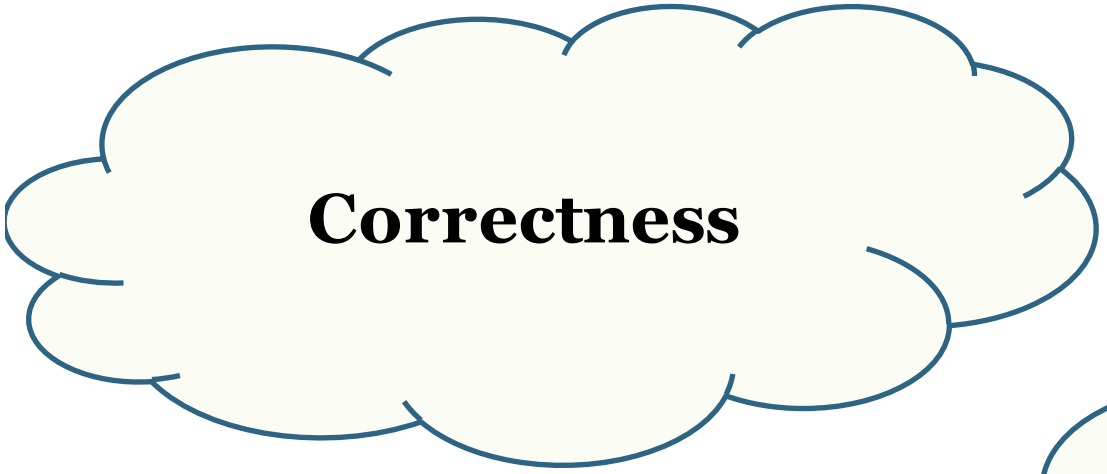
REQUIRED CHARACTERISTICS

Scripting languages that are appropriate for skilled and unskilled programmers.

Systems will need to have automated algorithms with human help only when necessary.

New data sources must be integrated incrementally as they are uncovered.

CHALLENGES



REQUIREMENTS

- Datasets:
  - Training data
  - Clean data
  - Test data
- Rules and constraints to detect dirty cells.
- Machine learning architecture: this may include
  - Clustering algorithm to detect outliers, dirty cells. For instance ActiveClean, Tamr.
  - Neural network based algorithm which is trained on a feature graph model to generate potential domain, for instance, HoloClean.
  - Classification and boosting algorithm (SVM, Naïve Bais etc.) to assign the correct class label from the domain based on a loss minimization function or to detect duplicates, for instance, BoostClean and Tamr.

REQUIREMENT ENGINEERING FOR DATABASES AND DATA CLEANING

PRESENTED TO: PROF. DAN BERRY
PRESENTED BY: MARIAN BOKTOR
COURSE CODE: CS846

UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS
David R. Cheriton School
of Computer Science

How to know if a database is any good?

- **Helps support and ensure the accuracy and integrity of information**
- Divides information into subject-based tables to reduce redundant data
- Provides Access with the information it requires to join the information in the tables together as needed
- Accommodates your data processing and reporting needs
- It is easy to modify and maintain without affecting other fields or tables in the database
- Information is easy to retrieve, and user applications are easy to develop and build

How to know if a database is any good?

- The database is scalable, meaning that it can be expanded to meet the changing needs of an organization
- Normalized, to minimize data redundancy, I/O redesign transaction sizes and to enforce referencing integrity
- Semantically same data have same representations in heterogeneous sources
- We have to have a single view and access to accurate and consistent data
- Quality Design: data is stored in a single logical unit instead of multiple files, maintaining data integrity and security, and finally increase the performance of the database

How to know if database software is any good?

- Easy to use
- Customizable
- Mobile-optimized
- Real-time insights
- Cloud-based
- Tailored for any company
- Multiple database formats
- Define constraints to maintain data integrity
- e.g. TeamDesk, Knack, Oracle Database Cloud Service, Microsoft Azure Cloud

Why we need a good database?

- Centralized systems
- Better management of human resource (HR) matters
- Managing customer data and relationships
- Efficient inventory tracking
- Planning for growth

High Quality Data | Criteria

- Validity
  e.g. Mandatory, Data-type, Range, Foreign-key and Unique constraints
- Accuracy
- Completeness
- Consistency
- Uniformity
- Traceability
- Timeliness

Prototyping & Testing

Prototype Validation and Evaluation:

- Assessing different models performance on predefined quality metrics.
- Comparing performance of different models.
- Hyperparameter tuning : performing model-specific optimizations.
- Checking whether the predictions make sense when comparing to ground truth.
- Are the results significant enough to make an impact on the present business situation?
- Do we require any additional features/data that can help in further improving the performance?
- Brainstorming with team.

Evaluation Metrics

Why we evaluate the predictive performance of a model?

- to estimate the generalization performance, the predictive performance of our model on future (unseen) data.
- to increase the predictive performance by tweaking the learning algorithm and selecting the best performing model.
- to identify the machine learning algorithm that is best-suited for the problem at hand by comparing different algorithms.
- to know when to update the model.

Offline Evaluation

measures offline metrics of the prototyped model on historical data like accuracy or precision recall.

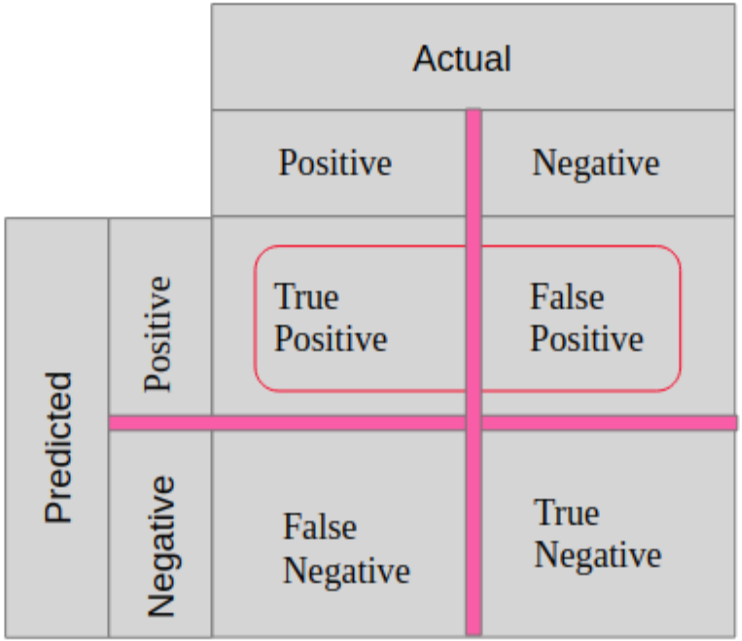
Online Evaluation

might measure business metrics such as customer lifetime value, which may not be available on historical data but are closer to what your business really cares about.

Confusion Matrix

Precision = TP / (TP + FP)
= TP / Total Predicted Positive

- Proportion of data points that the model says are relevant and are actually relevant.
- A good measure to determine, when the costs of False Positive is high.
- For eg, in email spam detection, an email that is not Spam (actually negative) has been predicted as spam by model.

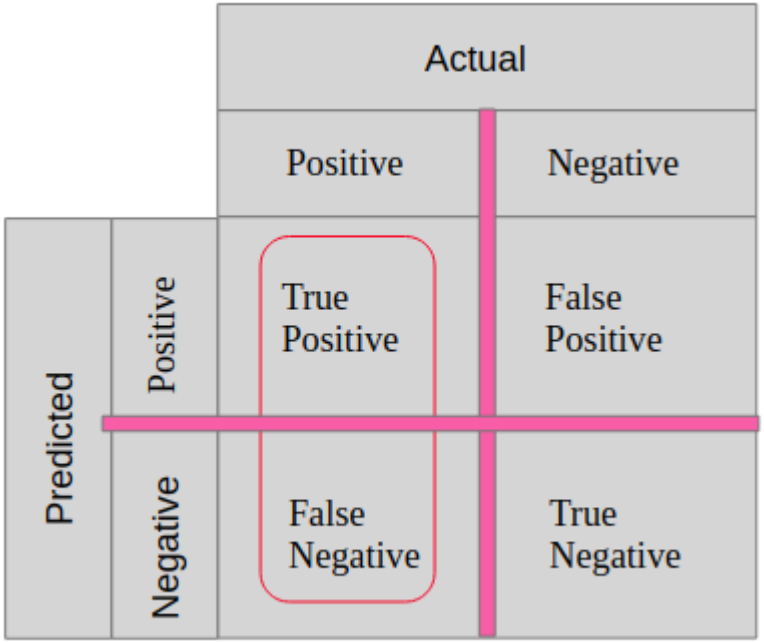


Confusion Matrix

Recall = TP / (TP + FN)

= TP / Total Actual Positive

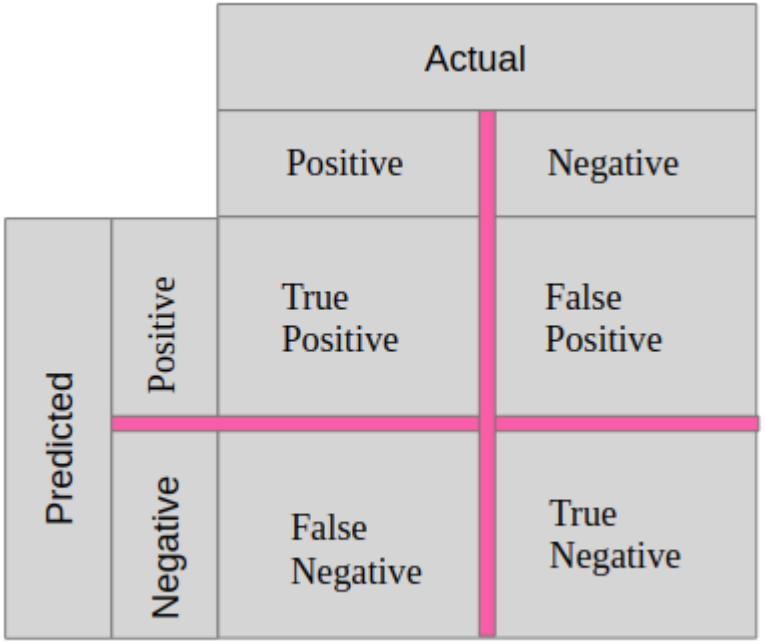
- Out of all the data points that are truly relevant in the dataset,how many are found by the model.
- A good measure to determine, when the costs of False Negative is high.
- For eg, in fraud detection, if a fraudulent transaction (actual positive) is predicted as non-fraudulent (predicted negative), can have bad outcomes.



Confusion Matrix

F1 Score = 2 * (Precision*Recall)
(Precision+Recall)

- Is the harmonic mean of precision and recall.
- Used when we want to seek a balance between Precision and Recall.
- Gives equal weight to both measures and is a specific example of the general Fβ metric where β can be adjusted to give more weight to either recall or precision



9. Evaluation Metric

- Evaluating your machine learning algorithm is an essential part of any project
- Assess the quality of the model
- Depends on:
  - Outcome of the project
  - Problem statement
  - Dataset at hand
- Different metric for regression and classification problems

Metrics for Regression

- **Mean Absolute Error (MAE)** - average of the absolute differences between the prediction and actual values

$$MeanAbsoluteError = \frac{1}{N}\sum_{j=1}^{N}|y_j - \hat{y}_j|$$

- Gives an idea of the magnitude of the error, but no idea of the direction
- Example : House Price Prediction

Metrics for Regression

- **Mean Square Error (MSE)** - average of the square differences between the prediction and actual values

$$MeanSquaredError = \frac{1}{N}\sum_{j=1}^{N}(y_j - \hat{y}_j)^2$$

- **Root Mean Square Error (RMSE) :** Taking root of MSE and converts the units back to the original units of the output variable

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(p_i - a_i)^2}{n}}$$

- Example : House Price Prediction

Metrics for Regression

- **R Squared** - provides an indication of the goodness of fit of a set of predictions to the actual values. Also, called the coefficient of determination

$$\textit{Coefficient of Determination} \rightarrow \quad R^2 = \frac{SSR}{SST} = 1 - \frac{SSE}{SST}$$

$$\textit{Sum of Squares Total} \rightarrow \quad SST = \sum (y - \bar{y})^2$$

$$\textit{Sum of Squares Regression} \rightarrow \quad SSR = \sum (y' - \bar{y'})^2$$

$$\textit{Sum of Squares Error} \rightarrow \quad SSE = \sum (y - y')^2$$

- Example : House Price Prediction

Metrics for Classification

- **Accuracy** - number of correct predictions made as a ratio of all predictions made

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

- Works well only if there are equal number of samples belonging to each class.
- Example : Classify email spam or not spam

Metrics for Classification

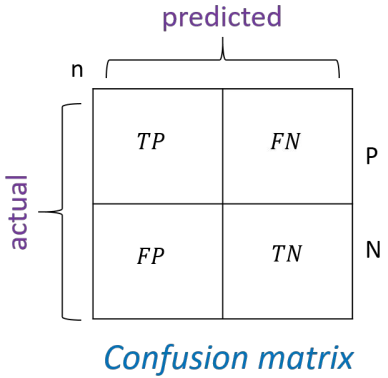
- **Log Loss** - classifier must assign probability to each class for all the samples

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

- The scalar probability between 0 and 1 can be seen as a measure of confidence for a prediction by an algorithm.
- Example : Classify a set of images of fruits which may be oranges, apples, or pears.

Metrics for Classification

- **Confusion Matrix**- number of correct and incorrect predictions made by the classification model compared to the actual outcomes in the data



Confusion matrix

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

- Used for imbalanced class

Metrics for Classification

- **Area Under the Curve(AUC)**- represents a model's ability to discriminate between positive and negative classes.
- Performance metric for binary classification

$$\frac{\mathrm{TP}}{P} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \qquad \frac{\mathrm{FP}}{N} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

- An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model as good as random
- Used for imbalanced classnvbv

Metrics for Classification

- **F1 Score** - Harmonic Mean between precision and recall. tell sow precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

$$F_1 = \left(\frac{\mathrm{recall}^{-1} + \mathrm{precision}^{-1}}{2} \right)^{-1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

- Range from [0, 1]
- F1 Score tries to find the balance between precision and recall

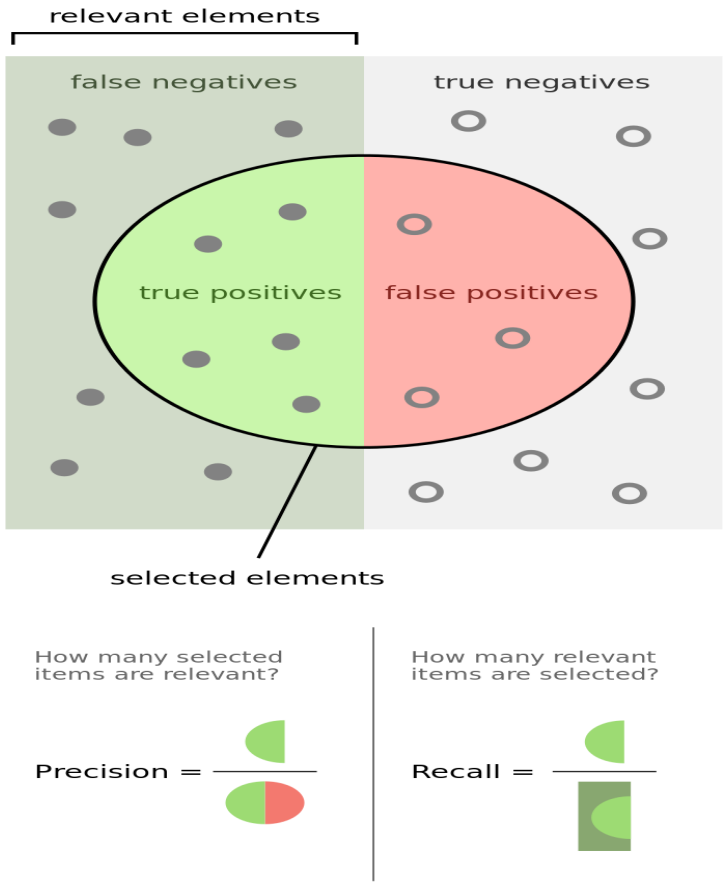
10. Formalism

Example: ML Model that predicts which tweets will get retweets

- **Task** (*T*): Classify a tweet that has not been published as going to get retweets or not.
- **Experience** (*E*): A corpus of tweets for an account where some have retweets and some do not.
- **Performance** (*P*): Classification accuracy, the number of tweets predicted correctly out of all tweets considered as a percentage.

REQUIREMENTS

- Evaluation metrics:
  - Precision
  - Recall
  - Accuracy (sometimes)
  - F1 score (sometimes)



EVALUATION

- The cleaned data test is matched to clean data that is prepared by a bunch of experts. The data is evaluated on:
  - Precision
  - Recall,
  - Accuracy (sometimes),
  - F1 score (sometimes).

EVALUATION

	Aggregator	Data Tamer
Total records	146690	
Pairs reported as duplicates	7668	180445
Common reported pairs	5437	
Total number of true duplicates (estimated)	182453	
Reported true duplicates (estimated)	7444	180445
Precision	97%	100%
Recall	4%	98.9%

Figure 2: Quality results of entity consolidation for the web aggregator data

EVALUATION

- HoloClean Evaluation:
  - Evaluate on different datasets like hospital, flights, food, physicians.
  - On average the precision is 0.895,
  - On average the Recall is 0.765,
  - On average the F1 Score is 0.819.

EVALUATION

BoostClean achieves up to 81% accuracy and is competitive with hand-written rules, and the word embedding features significantly improve the detector accuracy.