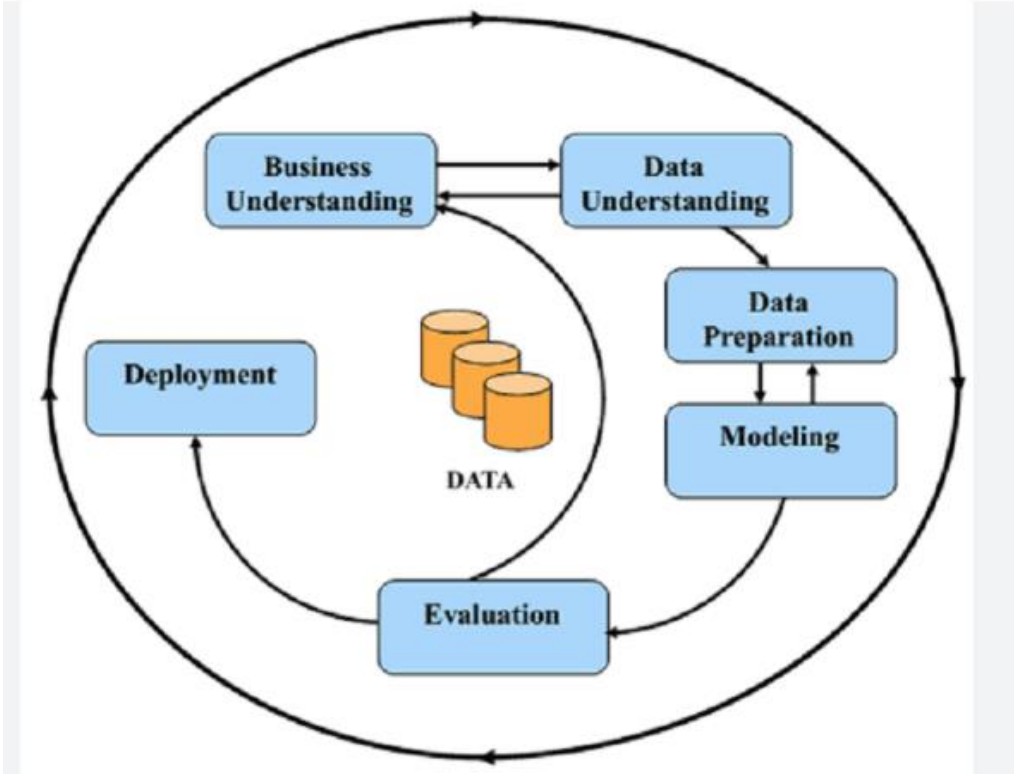
Churn Case Study



Business problem: what is exactly the problem we should solve

- Define churn clearly: some options- subscription cancellation, inactivity for 30+ days, no purchase in 90 days.
- Decide on actionability: how much in advance do we want to predict
- What are we going to do with the results
- Frequency of predictions: model should be run daily, weekly, monthly?

ML problem formulation

- Type of learning
- Target variable
- Prediction window & feature lookback window
- What is needed as a result: labels or scores or ranking
- Population to which is applied (maybe: customers who have been with the company at least x time)
- Which errors are more costly: false positives or false negatives

Data understanding

- Historical data availability ?
- Possible features:
  - Customer Features (static)
    - Demographics: age, region, income bracket
    - Account Info: signup date, subscription type, channel of acquisition
  - Usage Features (dynamic/behavioral)
    - Frequency: Number of logins, purchases, sessions in last x days
    - Recency: when was the service last used
    - Monetary: Total spend in recent periods
  - Product Interaction
    - Changes in plan/tier
    - Complaints or refunds
  - Presence of a competitor in the area
- Dealing with imbalance: subsample non churners? include churners from longer time period?

Feature engineering

- What makes a good feature?
- Will the feature be available at prediction time (data leakage)
- Instead of absolute values of usage or spending, aggregations: weekly, monthly?
  - averages:
  - delta in usage:
  - rate of change
  - ratios (refunds to purchase)

- Instead of timestamps:
  - recency values: days since last complaint, last usage?

- Competitor presence
  - just a flag (yes/no), more granular (offers better or same service)

EDA

- Verifications of data quality
  - do values/distributions “make sense”
  - missing values (random, data entry errors?)

- Categorical values:
  - high cardinality: aggregation (service levels into higher bundles, less frequent categories into “other”, cities into regions or lat/long coordinates)

Modeling

- Train validation test split: time sensitive
- Imbalanced data treatment
- Typical steps:
  - missing value
  - categorical feature encoding
  - scaling
  - data balancing
  - model fitting
- Use different models and tune parameters, think of the optimization metric, joint optimization (optimize over all pipeline steps at the same time, not sequentially): number of combinations to test vs computation time
- If different approaches are executed in different pipelines, are they evaluated on the same data, so performance is comparable: validation dataset

Modeling in practice

- Incremental approach:
  - try a smaller time period for feature engineering, do model selection and tuning, evaluate on validation
  - add longer data from time period, do model selection and tuning, evaluate on validation
  - try usage metrics, and then usage + demographics, and compare

Interpretability

- What are the features important to the model (do they align with business intuition? Not necessary, sometimes new patterns can be learnt)
- What is the impact of these features

Evaluation

- Baseline for comparison: consider those clients “who decreased their usage x%” as future churners?
- Evaluate all metrics on a test set not used for anything else before
- What type of errors is model making? Over or underestimating churners?
- Are errors more present in one service/age group…

Deployment

- Once the model is put in production, how do we know is it any good? (A/B testing)
- How often should we retrain? (data drift, score distribution)