A Practical Introduction to Data Science

Part 3

Introduction to Supervised Learning

Gergely Zsombor Haász

haasz.zsombi@gmail.com

Course Agenda

The ML Development Pipeline

EDA

Preprocessing

- Target and Sample Preprocessing
  - Target Definition
  - Sampling and Splitting
- Feature Preprocessing
  - Feature Cleaning
  - Feature Engineering
  - Feature Encoding
  - Feature Selection

Model Building

- Training a Baseline Model
- Training Challenger Models
  - Hyperparameter optimization
  - Cross-Validation

Evaluation

- Performance Evaluation
- Calibration Testing
- Explainability

Deployment & Monitoring

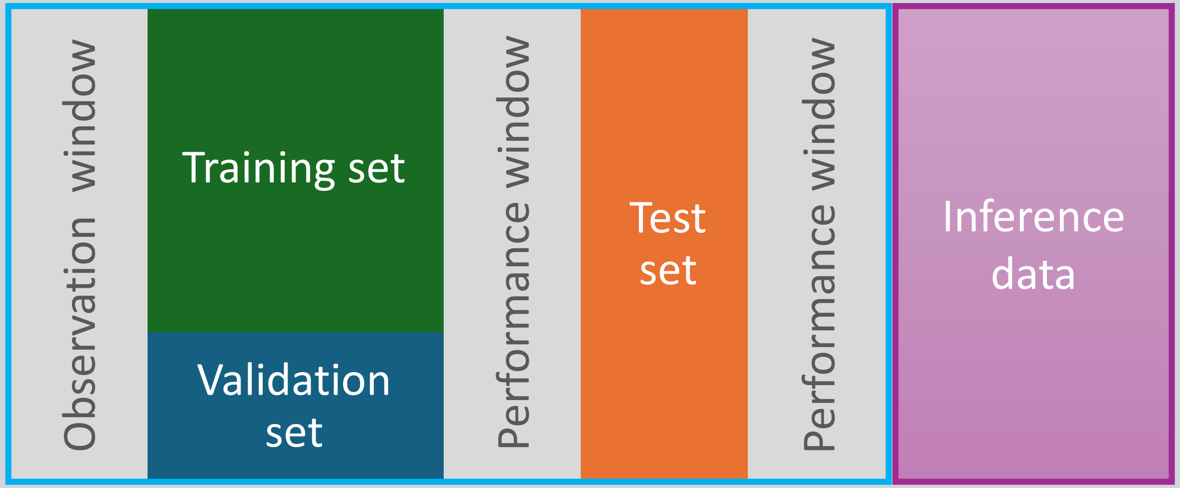
Target Definition

Challenges of Data Labelling:

1. Reliability
    - E.g. images, fraud
2. Cost and time of labeling
    - E.g. (medical) images
3. Measurability
    - E.g. customer satisfaction, fraud
4. Multiple Definitions or Proxies
    - Default or fraud definition
    - Performance Window trade-offs (number of cases, time delay, business need)
5. Class Imbalance
    - Undersampling or Oversampling
    - Different Proxy
    - Class weights
    - Model selection

Sampling and Splitting

- The goal is future inference (generalization)
- Choosing Train, Validation and Test Samples
  - Not too old, but not too small
  - Consider economic cycles, yearly seasonality etc.
  - Random split and out-of-time test
- Time Periods and Snapshots
- Observation Window
- Performance Window
- Data Leakage
  - Temporal leak
  - Grouped/redundant data
  - Preprocessing



Feature Cleaning

Missing Values

- Consider business meaning
- Removing columns or rows
- Filling Missing Values: zero, mean, mode, separate category
- Avoid imputation leakage

Outliers

- Remove or Replace
- Feature Engineering
- Choice of Algorithm

Feature Engineering and Encoding

- **Aggregation over time**
  - Time since last event
  - Frequency (number of events)
  - Total/average value during the period
  - Time or frequency of change
  - Stability, trend
- **Calculated variables**
  - Distance or Similarity
  - Diversity
  - Ratios instead of absolute values
- **Feature crosses**

- **Numerical features:**
  - Binning and clipping
  - Logarithmic transformation
  - Min-Max Normalization
  - Standard Scaling
- **Categorical features:**
  - Merging (categorical)
  - One-hot encoding (dummy variables)
  - Weight of evidence transformation
  - Target encoding

Feature Selection

Too many features increase complexity and add noise

1. Initial feature selection:
   - Business meaning
   - Target leakage
   - Predictive power
   - Multicollinearity
   - Stability
2. Training-time feature selection *(discussed in the following lecture)*

The ML Development Pipeline

EDA

Preprocessing

- Target and Sample Preprocessing
    - Target Definition
    - Sampling and Splitting
- Feature Preprocessing
    - Feature Cleaning
    - Feature Engineering
    - Feature Encoding
    - Feature Selection

Model Building

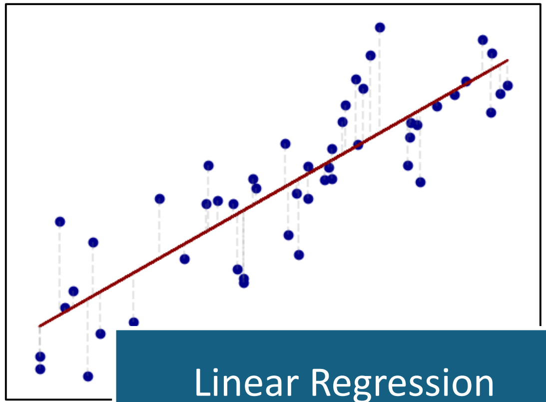
- Training a Baseline Model
- Training Challenger Models
    - Hyperparameter optimization
    - Cross-Validation

Evaluation

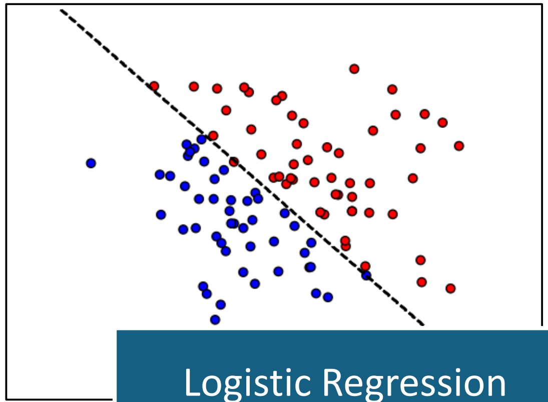
- Performance Evaluation
- Calibration Testing
- Explainability

Deployment & Monitoring

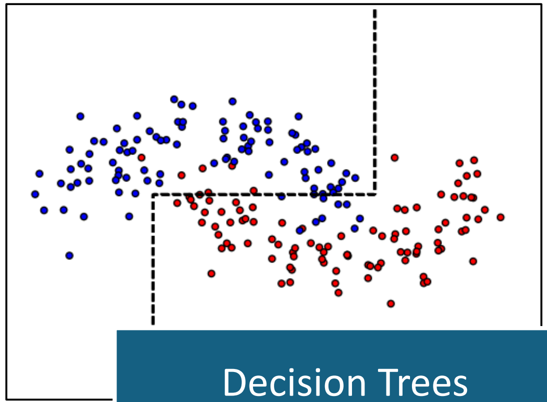
Fundamental Models



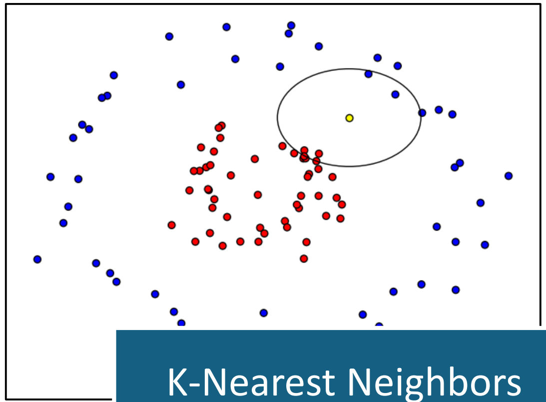
Linear Regression



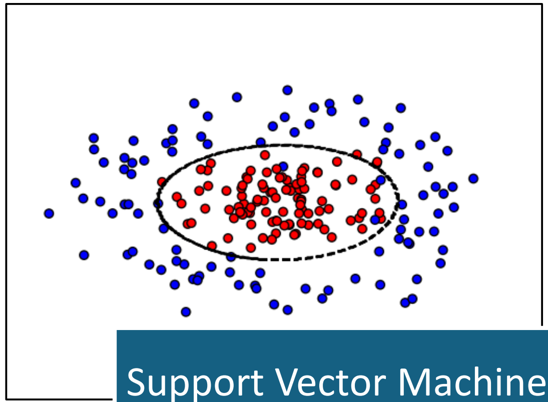
Logistic Regression



Decision Trees



K-Nearest Neighbors



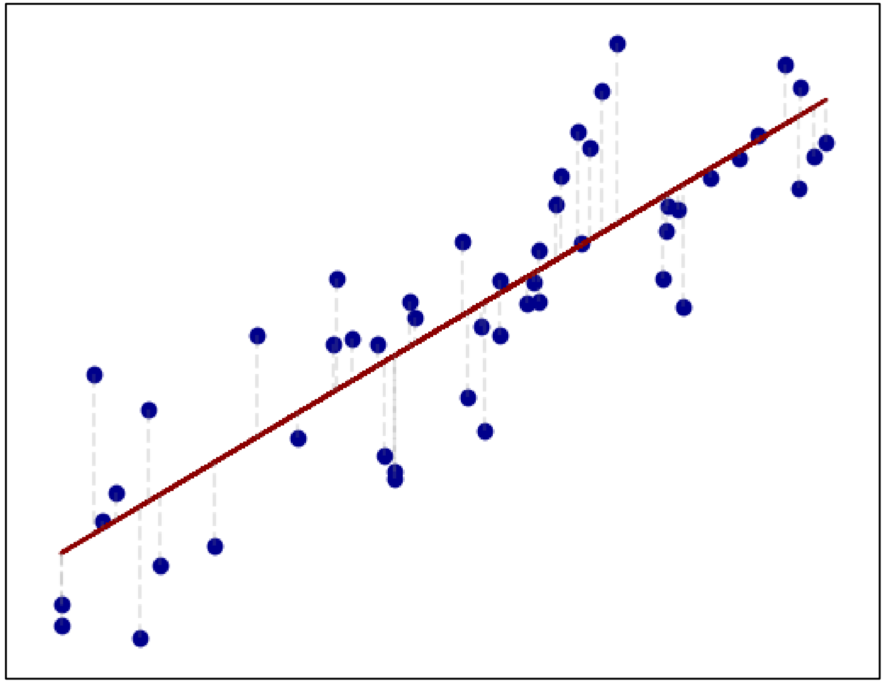
Support Vector Machines

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Naive Bayes

Linear Regression

- Isaac Newton, 1700
- Continuous target variable
- Assumes linear relationship between input features (predictors) and the target variable
- Model training: estimate the coefficients (betas) to minimize the loss
- OLS minimizes the sum of the squared differences between observed and predicted values.



$$Y = \beta_0 + \sum_{i=1}^{p} \beta_i X_i + \epsilon_i$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Linear Regression

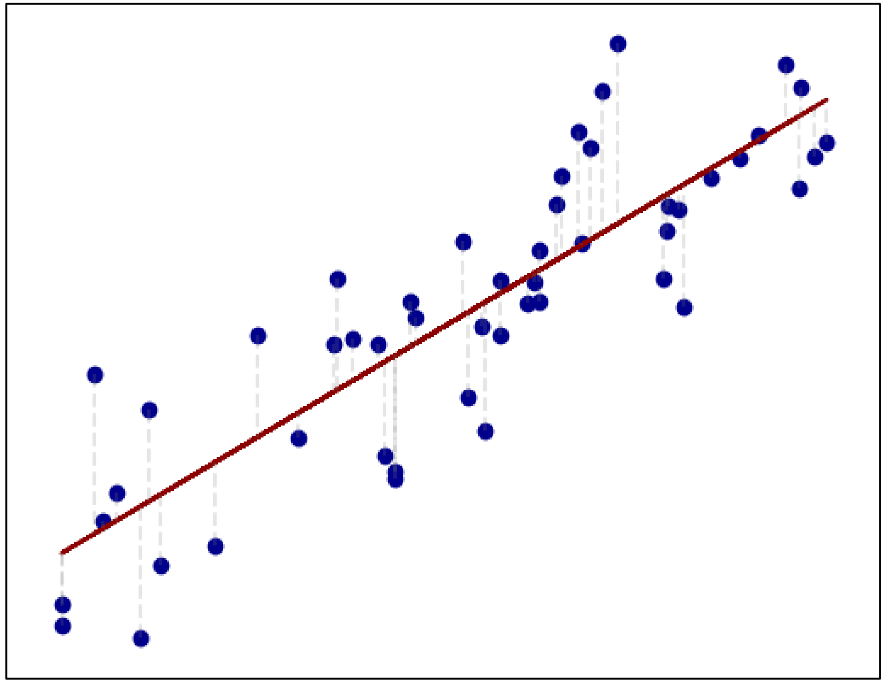
- Model coefficients (betas) are easy to interpret
- Simple model
- Cannot recognize complex patterns
- Sensitive to outliers
- Coefficient of determination (R^2) – explained variance
- Assumptions:
  - Linear relationship
  - No multicollinearity
  - Residuals:
    - Normality
    - No autocorrelation
    - Homoskedasticity

$$SST = \sum_{i=1}^{n}(y_i - \bar{y}_i)^2$$

$$SSR = \sum_{i=1}^{n}(\hat{y}_i - \bar{y}_i)^2$$

$$SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

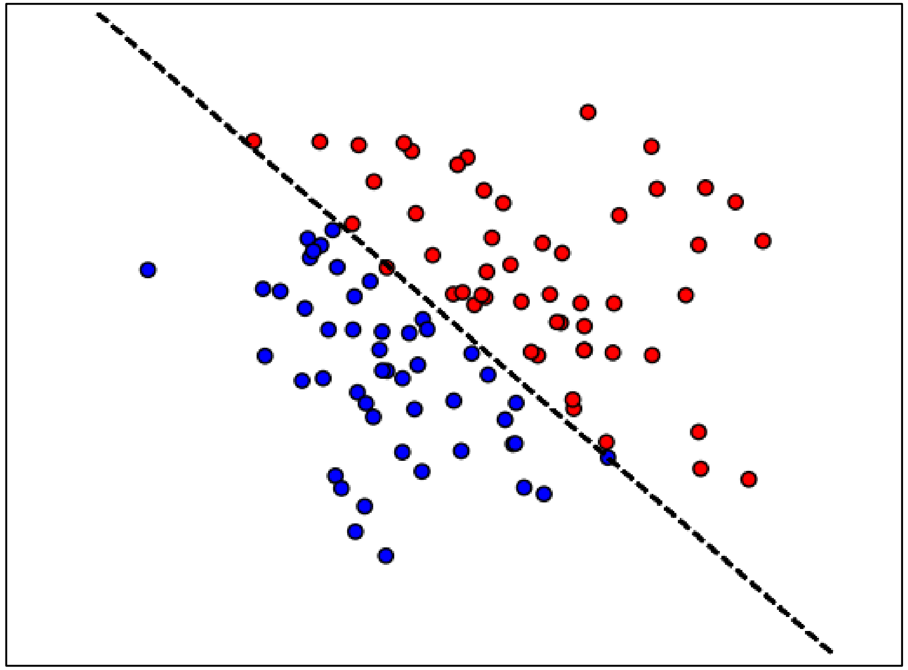
$$R^2 = 1 - \frac{SSE}{SST} = \frac{SSR}{SST}$$



Logistic Regression

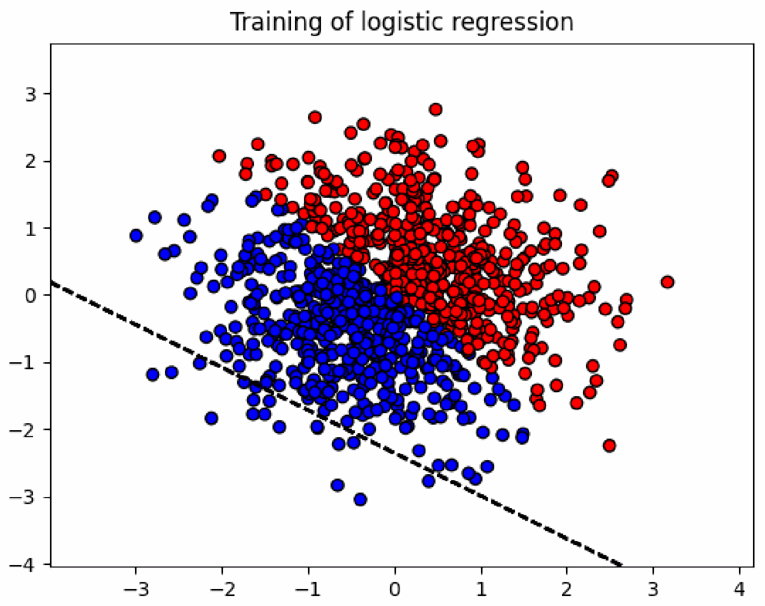
- Based on linear regression, adjusted for classification
- Binary Target: $\mathrm{Y} \in \{0, 1\}$
- Model Output: $\mathrm{P}(Y = 1 \mid X) \in (0, 1)$
- Linear equation: $Z = \beta_0 + \sum_{i=1}^{p} \beta_i X_i$
- Sigmoid transformation: $\mathrm{P} = \frac{1}{1+e^{-Z}}$
- Binary Output: $\hat{Y} = I(P > cutoff)$
- Z is called log odds or logit

$$Z = logit(P) = \log\left(\frac{P}{1 - P}\right)$$



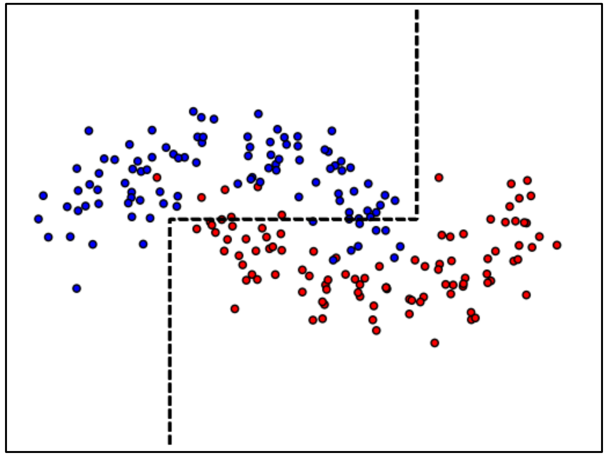
Logistic Regression

- Model training: estimate the coefficients (betas) to minimize the loss
- $\text{Logloss} = -\frac{1}{N}\sum_{i=1}^{N}(Y_i \ln P_i + (1 - Y_i)\ln(1 - P_i))$
- Linear model: The decision boundary is a hyperplane
- Model coefficients (betas) are easy to interpret
- $P(rain) = \text{sigmoid}(-1 + 2X_1 - 3X_2) = 0.8$
- Great benchmark and good for small data
- Less prone to overfitting
- Cannot recognize complex patterns

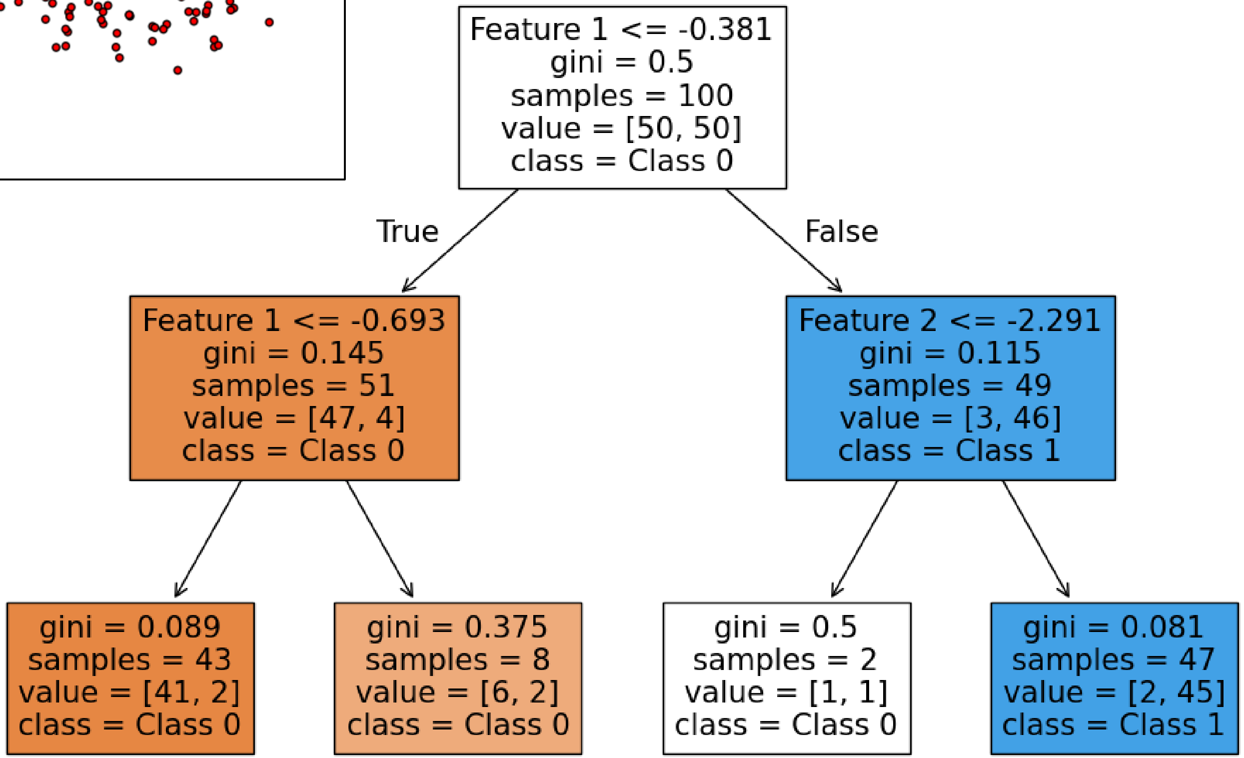


Decision Tree

- CART – 1984
- Non-linear model
- Splitting Criteria:
  Gini Impurity, Entropy
- Tree depth, leaf size
- Easy to understand and interpret
- Handles both numerical and categorical data
- Less sensitive to outliers
- Bad extrapolation
- Prone to overfitting

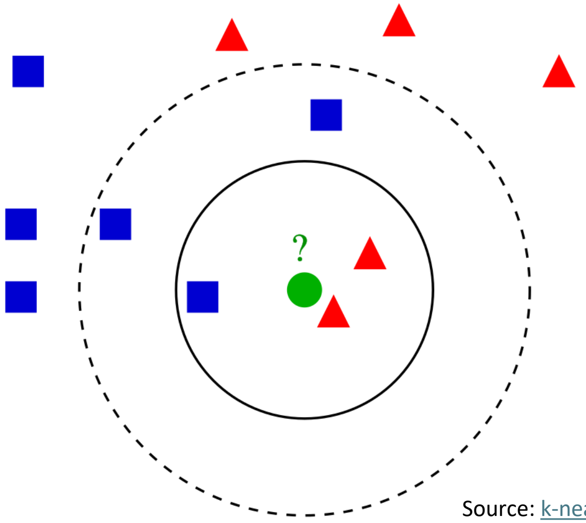


$$Gini = 1 - \sum_{i=1}^{k} P_i^2$$



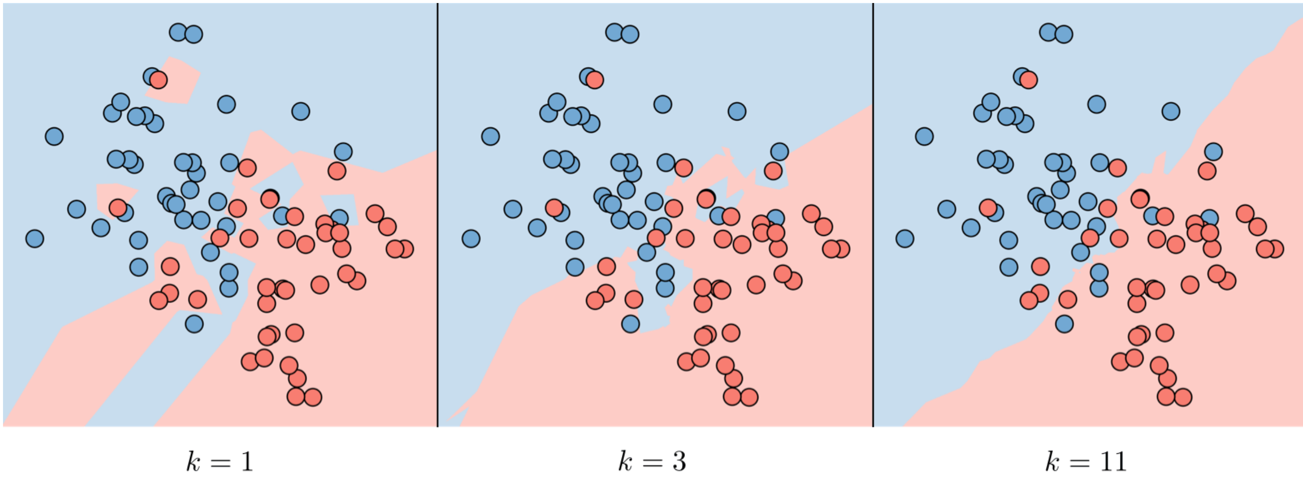
k-Nearest Neighbors

- Instance-based learning
- No explicit training phase, makes predictions using the entire dataset.
- Choose a distance metric (Euclidean, Manhattan)
- Choose k (number of neighbors)
- Simple and intuitive, for both classification and regression
- Computationally expensive
- Sensitive to irrelevant features



$$D(X,Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

Source: k-nearest neighbors algorithm – Wikipedia



Source: CS 221 - Reflex-based Models Cheatsheet

Naive Bayes Classifier

- Based on Bayes' Theorem
- Predicts the probability of a class given the input features
- Assumes conditional independence between predictor variables given the class
- Easy to implement and efficient on large datasets
- Used in text classification, spam filtering, sentiment analysis
- Limitation: independence assumption

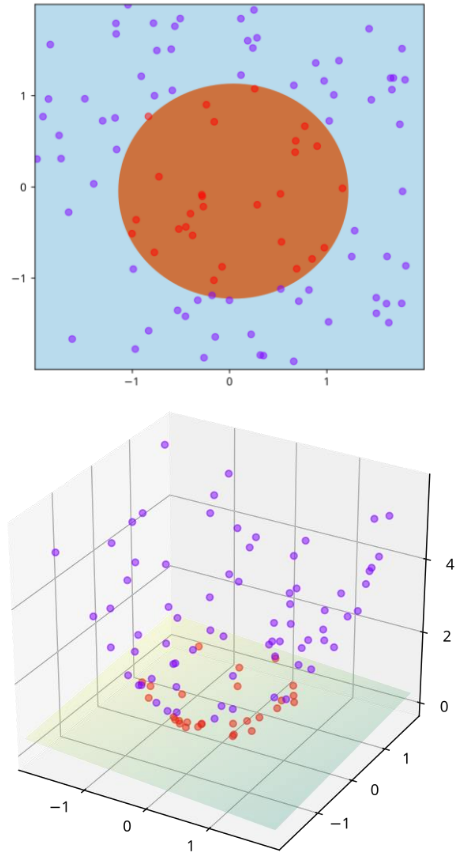
$$P(C|X) = \frac{P(X|C)P(C)}{P(X)}$$

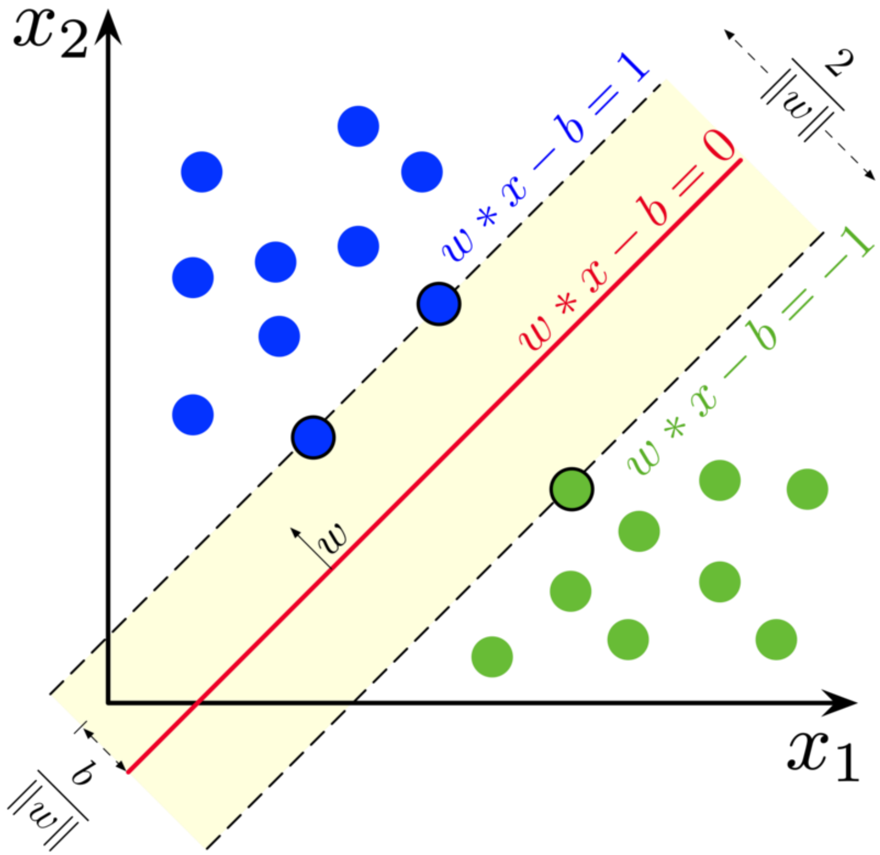
$$P(X|C) \approx P(X_1|C) \cdot P(X_2|C) \cdot \ldots \cdot P(X_n|C)$$

Support Vector Machines

1. Linear classifier with the maximum margin hyperplane
2. Soft margin (cost: C)
3. Nonlinear kernel

- Effective in complex and high-dimensional data
- Computationally expensive





Source: [Support vector machine – Wikipedia]

The ML Development Pipeline

EDA

Preprocessing

- Target and Sample Preprocessing
    - Target Definition
    - Sampling and Splitting
- Feature Preprocessing
    - Feature Cleaning
    - Feature Engineering
    - Feature Encoding
    - Feature Selection

Model Building

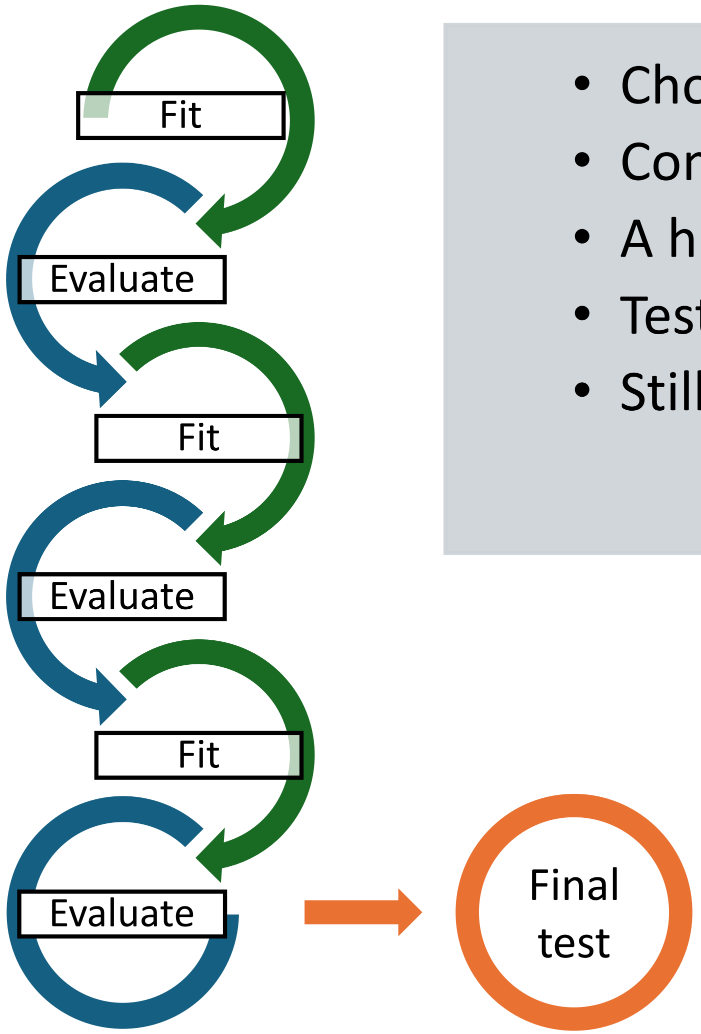
- Training a Baseline Model
- Training Challenger Models
    - Hyperparameter optimization
    - Cross-Validation

Evaluation

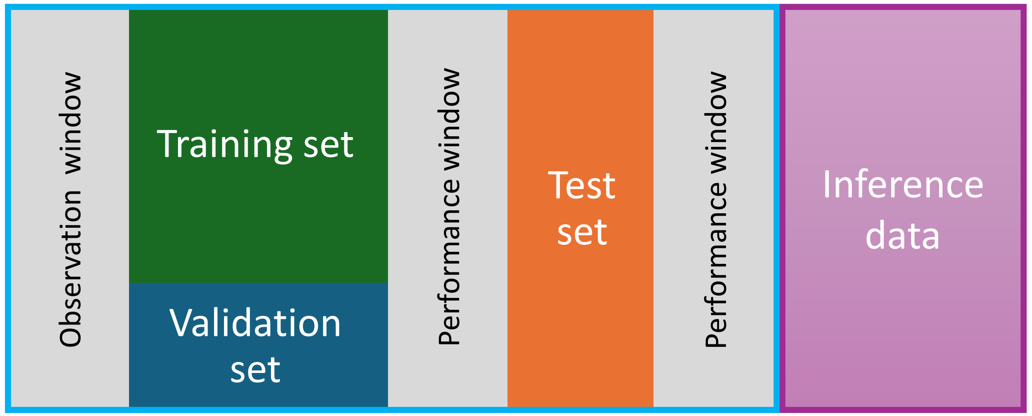
- Performance Evaluation
- Calibration Testing
- Explainability

Deployment & Monitoring

Model Performance Evaluation



- Choose proper evaluation metrics
- Compare performance on train/validation/test samples
- A huge performance drop indicates overfitting
- Test sample should be out-of-time
- Still just a proxy for the future inference performance



Regression Evaluation

- $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ penalizing large errors
- $\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$ simple average error
- $\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$ percentage, useful for comparison
- $R^2 = 1 - \frac{SSE}{SST} = \frac{SSR}{SST}$ explained variance
- Adjusted R^2 penalizing more predictors, useful for comparison
- Evaluate overall performance, performance within business segments and over time (if possible)

Classification Evaluation

- Binary Target: $\mathrm{Y} \in \{0,1\}$
- Model Output: $\mathrm{P}(Y = 1 \mid X) \in (0,1)$
- The estimated probabilities give a ranking of the test sample
- $\hat{y} = I(P > cutoff)$

Confusion matrix		Prediction	
		Negative	Positive
Observed Label	Negative	True Negatives	False Positives
	Positive	False Negatives	True Positives

Classification Evaluation

- $Accuracy = \frac{Correctly\ Classified\ observations}{All\ observations}$

- Which classification is better?

Accuracy = 98%		Prediction	
		Negative	Positive
Observed Label	Negative	1000	10
	Positive	10	0

Accuracy = 95.6%		Prediction	
		Negative	Positive
Observed Label	Negative	970	40
	Positive	5	5

Classification Evaluation

- $\text{Recall (TPR)} = \dfrac{\text{True Positives}}{\text{Actual positives}}$

- $\text{Precision} = \dfrac{\text{True Positives}}{\text{Predicted as positives}}$

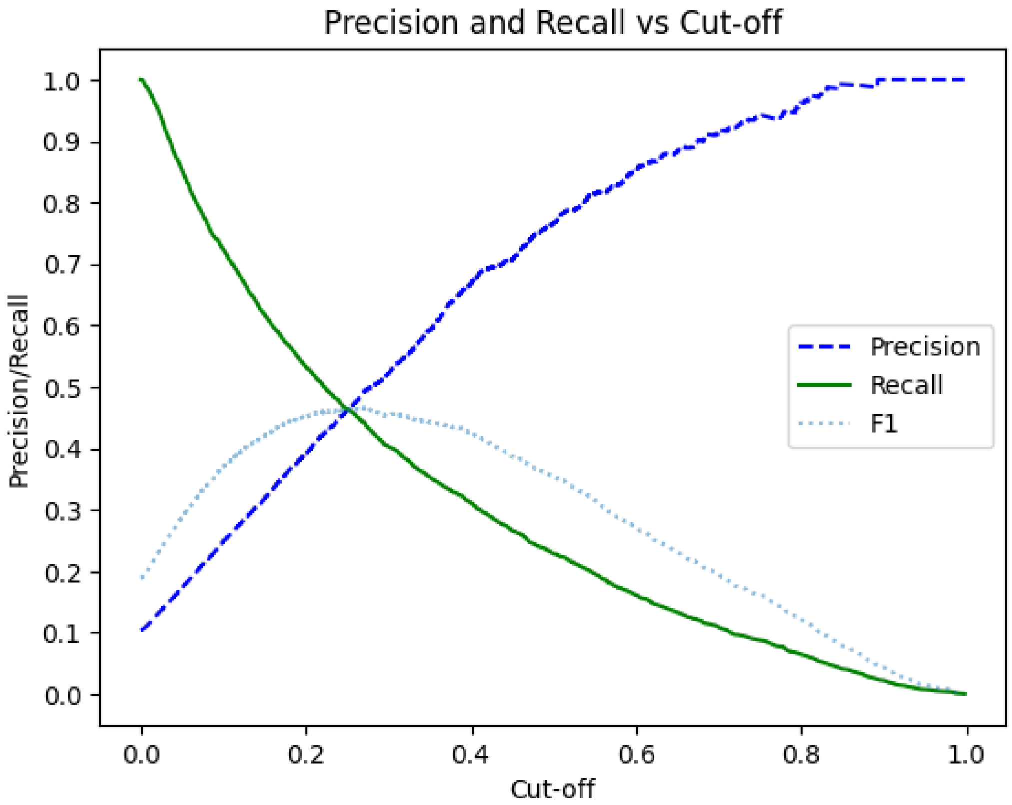
- $\text{F}_1 = 2 \cdot \dfrac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

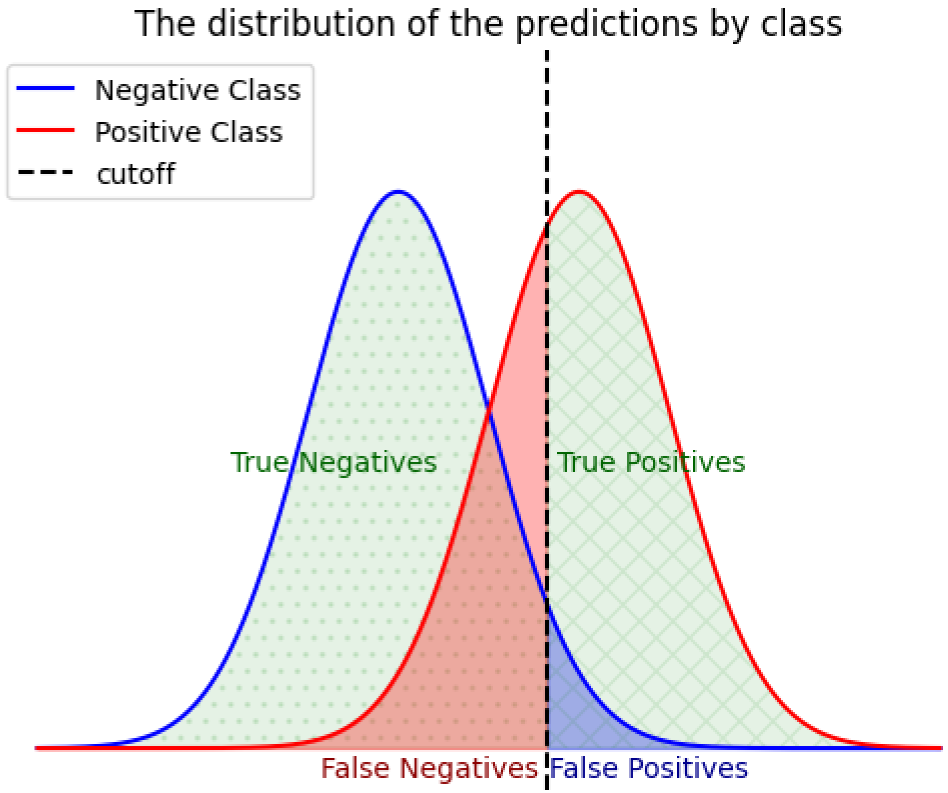
- Imbalanced data
- Different misclassification costs

Recall = 50% Precision = 11%		Prediction	
		Negative	Positive
Observed Label	Negative	960	40
	Positive	5	5

Classification Evaluation

- There is a trade-off between the two types of error. Which one is more important?
- Changing the cut-off influences these metrics but does not affect the ranking of the model!

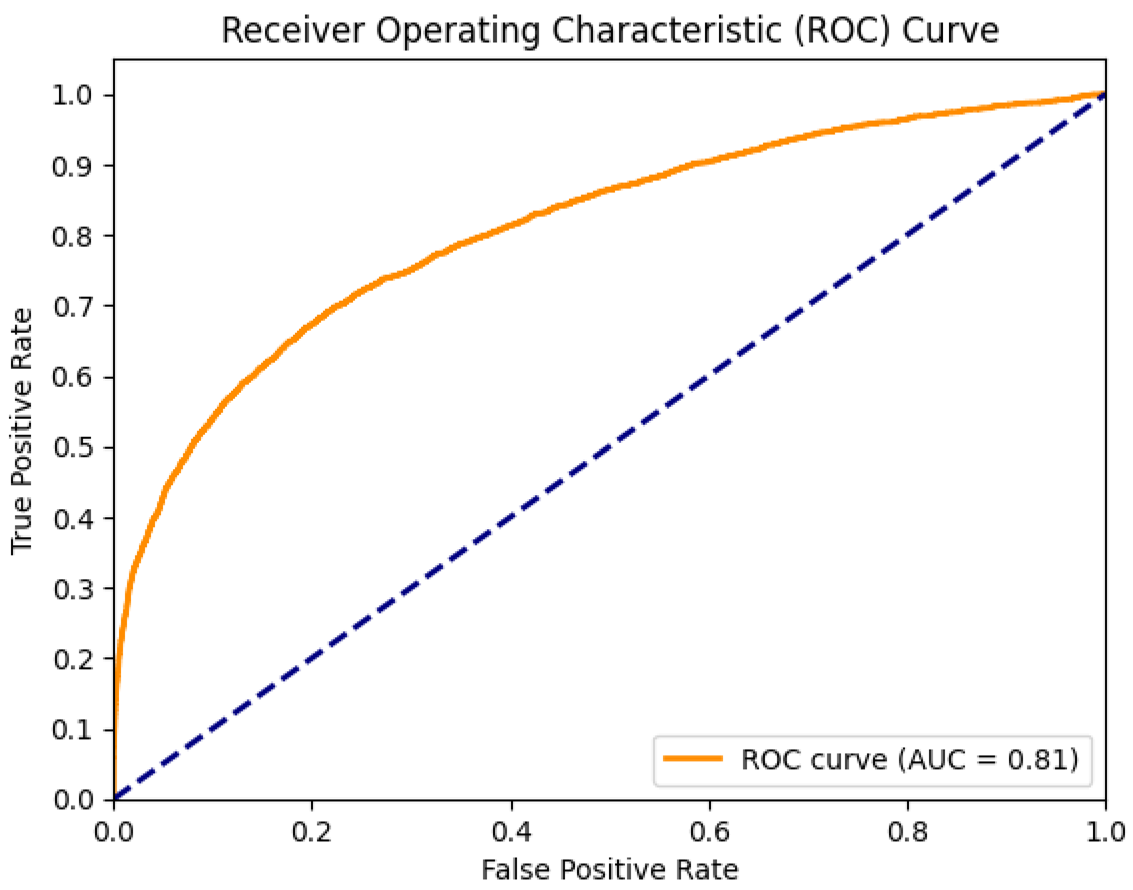




Classification Evaluation

ROC-AUC evaluates the ranking:

- Calculate and plot TPR vs FPR for every possible cut-off value
- AUC: area under the curve
- Not relying on the cut-off
- Good for comparison
- Random model: AUC ~ 0.5
- Perfect model: AUC = 1
- Use **PR-AUC** for imbalanced datasets (prioritizes positive class detection)



Classification Evaluation

1. Optimize the ranking
   - minimize the loss function during training,
   - evaluate and compare models based on ROC and AUC (or similar)
2. Choose the cut-off value
   - Consider the misclassification costs
   - Interpret results with the confusion matrix and derived metrics
3. Evaluate overall performance, performance within business segments and over time (if possible)

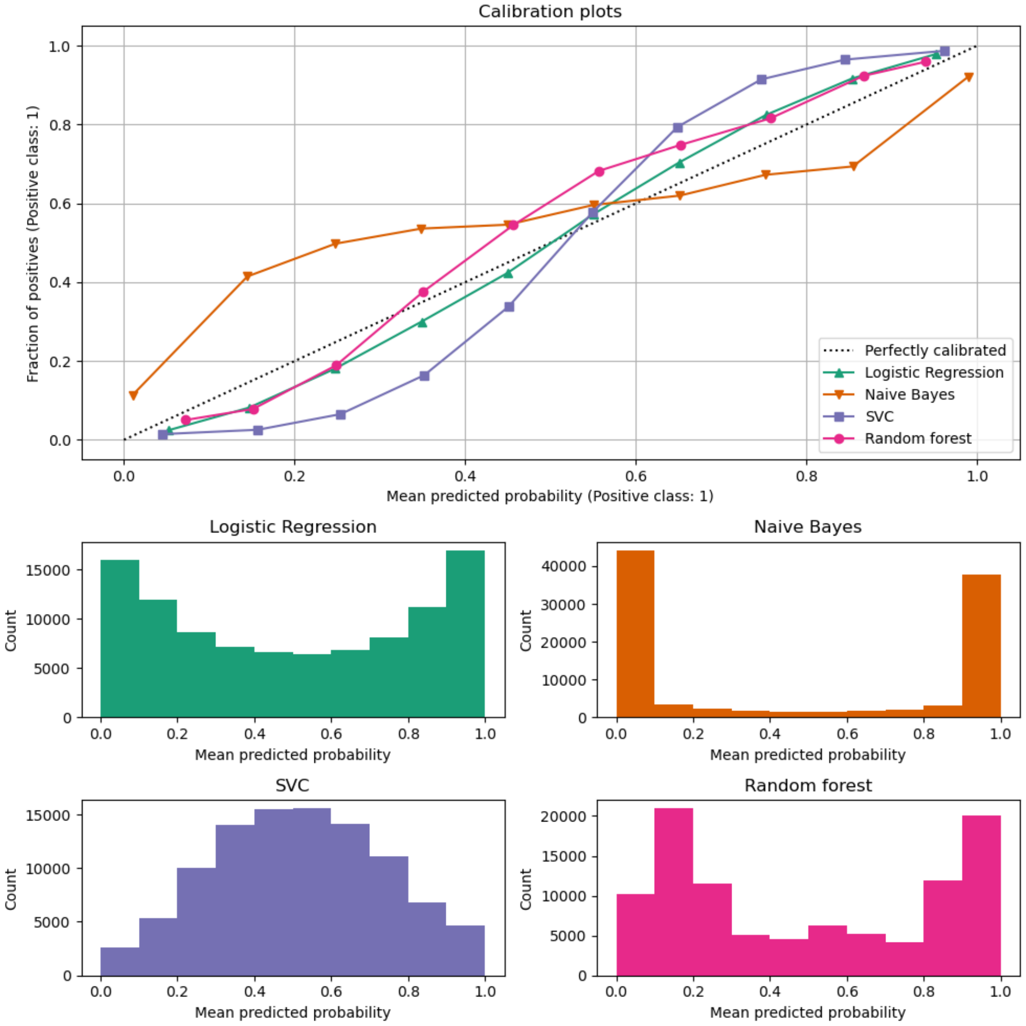
Calibration for Classification

- Predicted probabilities should match the actual probabilities observed in the data
- Methods:
  - Platt scaling: fit a logistic regression model to the scores
  - Scaling the log odds:

$$\log \text{odds} = \alpha + \log \text{odds}$$

$$\log \text{odds} = \frac{\log \text{odds}}{\mathrm{T}}$$

- Calibration of business segments and over time



Source: [1.16. Probability calibration — scikit-learn 1.6.1 documentation]

Business Impact Estimation

Examples:

- Saved subscriptions
- Market growth
- Profit, e.g.

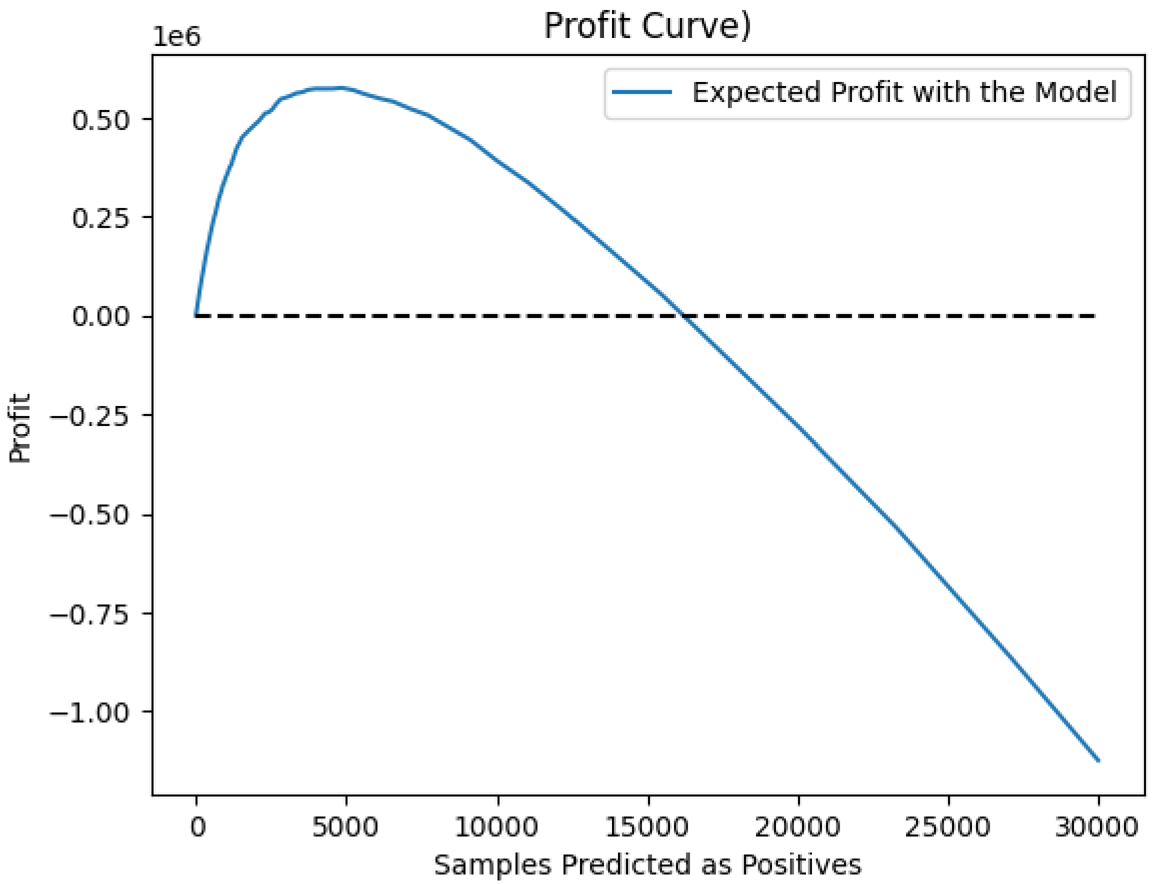
$$Profit = 500 * TP \ - 100 * FP$$

- Credit Risk Example:

$$EL = \sum(EAD * LGD * PD)$$

Cut-off choice:

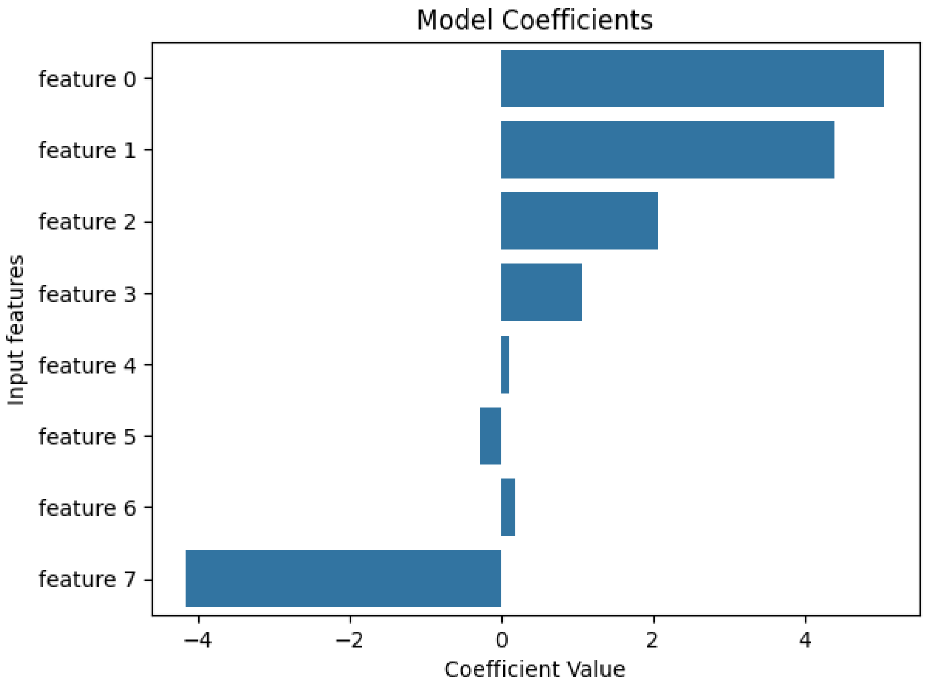
- Metric based
- Profit based
- Resource/business based (percentiles)



Explainability

As a model of a complex system becomes more complete, it becomes less understandable. (Bonini's paradox)

- Model Coefficients
- Feature Importances
- SHAP values
- Model Confidence and Out-of-Distribution Data



The ML Development Pipeline

EDA

Preprocessing

- Target and Sample Preprocessing
    - Target Definition
    - Sampling and Splitting
- Feature Preprocessing
    - Feature Cleaning
    - Feature Engineering
    - Feature Encoding
    - Feature Selection

Model Building

- Training a Baseline Model
- Training Challenger Models
    - Hyperparameter optimization
    - Cross-Validation

Evaluation

- Performance Evaluation
- Calibration Testing
- Explainability

Deployment & Monitoring

Thank you for your attention!

Your feedback would be much appreciated:

Any Questions?

Gergely Zsombor Haász

haasz.zsombi@gmail.com