IS4242
INTELLIGENT SYSTEMS & TECHNIQUES

L3 – Targeting Current Customers

Aditya Karanam

Announcements

- Programming Assignment – 1 will be released today
  - Due: September 12, 11:59 PM
  - Penalty for late submission, please start as early as possible

- Primers on Numpy and Pandas, and Visualization will be provided this week

In this Class

- Consumer Lifetime Value
- Logistic Regression
- Support Vector Machine
- Optimal threshold for classification in the context of marketing campaigns

Different Types of Marketing

- Mass marketing treats all customers as one group
- One-to-one marketing focuses on one customer at a time
- Target marketing to selected groups of customers or market segments
  - Lies between mass marketing and one-to-one marketing
- Target marketing involves direct marketing to those customers who are most likely to buy
  - Target marketing increases customer expenditures with the firm

Why Target Current Customers?

- Extracting profit from the existing customer is much easier than the acquiring a new customer
  - "Acquiring a new customer *can cost five to seven times* more than retaining an old one"
- Retaining is expensive as well
  - Mailings, phone calls, Google or Facebook targeting, etc.
  - Targeting and retaining *valuable* customer

Who is a target?

- A target is a customer who is worth pursuing

- Profitable customer – sales revenue from the target exceed costs of sales and support

- Customer with a positive lifetime value
  - Over the course of a company's relationship with the customer, more money comes into the business than goes out of the business

- How do we calculate Customer Lifetime Value (LTV)?

Customer Lifetime Value

- Lifetime value is the expected net *present value* of future profit contributions by the costumer
  - LTV $= \sum_{t=0}^{\infty} \frac{E(V_t)}{(1+\delta)^t} = \sum_{t=0}^{\infty} \frac{E(R_t - C_t)}{(1+\delta)^t}$
    - δ: discount rate
    - R_t & C_t represent revenue and cost from the consumer at t, respectively
    - Customer subscript is ignored in the above notation
- $E(V_t)$ depends on whether the consumer stays with the company until time *t*
- $E(V_t) = (R_t - C_t)P(Customer\ survives\ until\ t) = (R_t - C_t)S(t)$
  - $S(t)$ can be calculated based on hazard models

Customer Lifetime Value with Hazard Model

- Let T be a random variable representing the time customer attritions or leaves the company
  - $f(t)$: probability density function, $F(t)$: cumulative distribution function
- Let $S(t)$ be the probability that the customer attritions after time t
  - $S(t) = P(T > t) = 1 - F(t)$
- Hazard function: the probability that customer attritions during the instantaneous period Δt, given that the customer is with the firm up to t.
  - $h(t) = \frac{f(t)}{S(t)}$
- Retention rate at time t : $r(t) = 1 - h(t)$

Customer Lifetime Value: Geometric Distribution

- Most often, T is assumed to follow Geometric distribution
  - Measures the probability of success after a given number of trails
    - Success in this case is consumer leaving the company!
- $f(t) = p(1-p)^t$
- $S(t) = P(T > t)$
  - $\mathrm{P}(T > t) = \sum_{i=t+1}^{\infty} p(1-p)^{i-1} = (1-p)^t$
- $h(t) = p$
- $\mathrm{r}(t) = 1 - h(t) = 1 - p$ (a constant, hence t subscript is ignored)
  - $S(t) = \mathrm{r}^t$

Calculate the retention rate by

- LTV $= \sum_{t=0}^{\infty} \frac{E(V_t)}{(1+\delta)^t} = \sum_{t=0}^{\infty} \frac{(R_t - C_t)\mathrm{r}^t}{(1+\delta)^t}$

- Assuming constant revenue and cost across time,
  - LTV in the infinite horizon: $\frac{(R-C)(1+\delta)}{1+\delta-r}$

- LTV for a consumer: $\sum_{t=0}^{\infty} \frac{(R_t - C_t)\mathrm{r}^t}{(1+\delta)^t} = \sum_{t=0}^{\infty} \frac{m_t \mathrm{r}^t}{(1+\delta)^t}$
  - m_t: customer's profit contribution in time t

Increasing Marginal Revenues through Targeting

- By targeting current customers, we can improve marginal revenues in two ways: cross selling and upselling
- *Cross-selling*: Firms sell different products to its customers
  - For example, the customer uses Intuit's TurboTax software, and the company tries to sell the customer Quicken.
- *Up-selling*: selling "more" (higher volume, upgrades) of products they already are buying from the company
  - For example, a customer has $300,000 in term life insurance, and the company tries to sell the customer a $500,000 policy

Models for Targeting Current Customers

- Models that focus on what product the customer is likely to buy next
- Models that consider when the product is likely to be bought
- Our focus today: Models that consider how likely the customer is to respond to the cross-selling or up-selling offer
  - Sales promotions, Coupons, etc.
  - Techniques: Regression, Classification, etc.

Targeting Current Customers: Classification Techniques

Application: Modeling Response to Superstore Marketing

- A superstore is planning for the year-end sale.
- They want to launch a new offer - gold membership, that gives a 20% discount on all purchases.
- It will be valid only for existing customers and the campaign through phone calls is currently being planned for them
- The management feels that the best way to reduce the cost of the campaign is to make a predictive model to identify customers who might purchase the offer

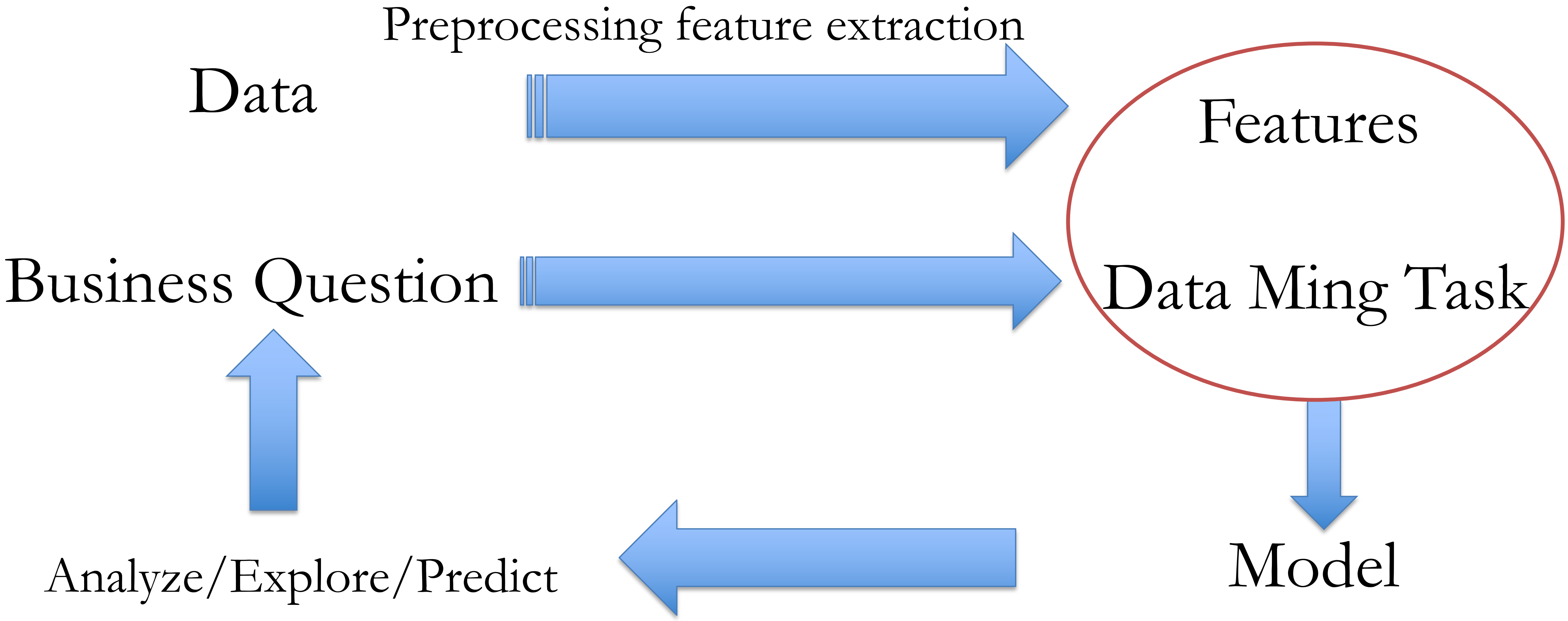
Data Description: Attributes

- **Outcome:** Response (target) - 1 if customer accepted the offer in the last campaign, 0 otherwise
- **Predictors:**
  - *Customer characteristics:* Year_Birth - Age, Education, Marital status, Income, Kidhome - number of small children, Teenhome - number of teenagers in customer's household
  - *Customer purchase behavior:* amount spent on fruits, fish and meat products, Sweets, Wines, Gold, No. of Catalog Purchases, Web purchases, Website visits, Deal purchases.
  - No of Complains, Recency, etc.

Application: Modeling House Prices

- Analyze the data to identify factors that impact response
- Build a prediction model to predict the probability of a customer will give a positive response.
  - Task: Classification
- The management will use this model to target customers through phone calls

Data Mining


Preprocessing feature extraction
Data
Features
Business Question
Data Ming Task
Model
Analyze/Explore/Predict

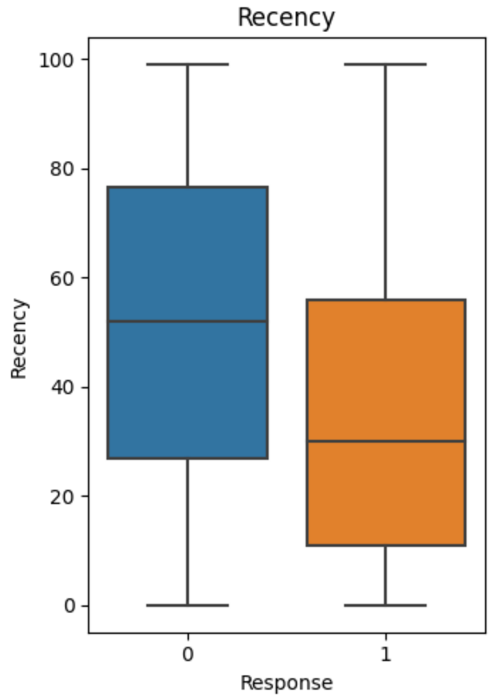
Classification Tasks

- Outcome: Response (binary variable)

- Simply way: look at the distribution of predictors for each class and try to identify the variables that have significantly different distribution

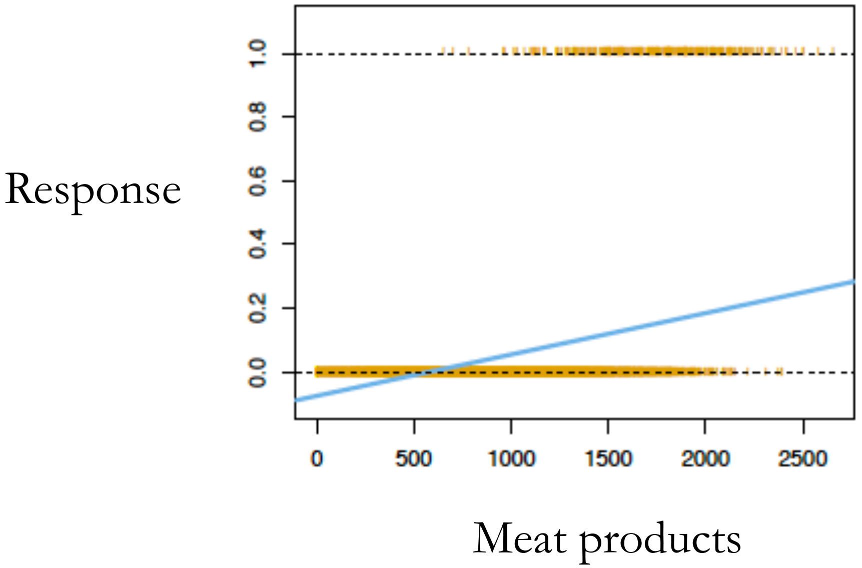
- This is good in identifying the variables that matter.
  - But doesn't provide predictions

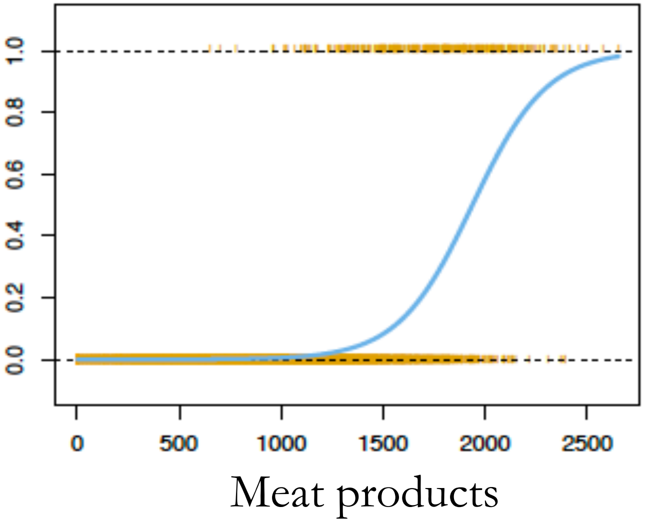
- Classification models: Logistic Regression and SVM



Regression?

- $P(\,response = Yes \mid predictor\ variables)$





- Left: linear regression → negative probabilities!
- Right: All probabilities lie between 0 and 1

Logistic Regression

- Odds: $\frac{P(Y)}{1-P(Y)} = e^{\beta_0+\beta_1 X_1+\ \ldots+\beta_p X_p}$

- Logit or log-odds: $\log\left(\frac{P(Y)}{1-P(Y)}\right) = \beta_0 + \beta_1 X_1 + \ \ldots + \beta_p X_p$

- Logistic function or sigmoid: lies between 0 and 1

- Unit increase in X_1 changes log-odds by β_1 or multiplies the odds by e^{β_1}

Estimating Regression Coefficients

- Approach: Maximum Likelihood Estimation

- Ex: Logistic regression with one variable: $\log\left(\frac{P(Y)}{1-P(Y)}\right) = \beta_0 + \beta_1 X_1$
  - $l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{j:y_i=0}(1 - p(x_j))$
  - Find coefficients that maximise likelihood

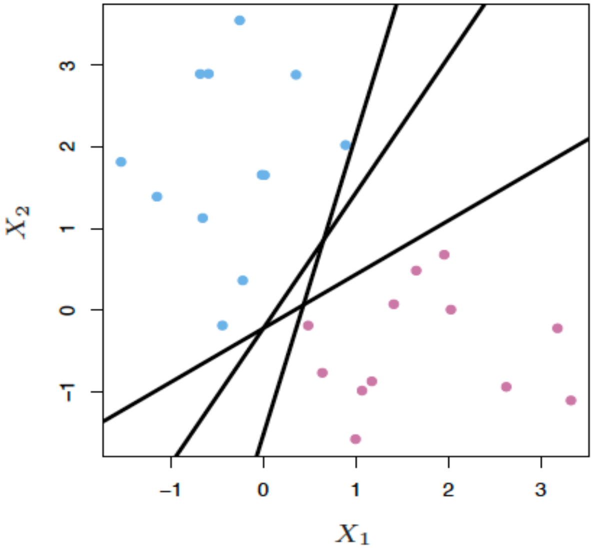
- $P(\widehat{y_i}) = \frac{e^{\widehat{\beta_0}+\widehat{\beta_1}X_1+ \ldots+\widehat{\beta_p}X_p}}{1+e^{\widehat{\beta_0}+\widehat{\beta_1}X_1+ \ldots+\widehat{\beta_p}X_p}}$
  - By default: $\widehat{y_i} = 1\ if\ P(\widehat{y_i}) > 0.5\ else\ 0$

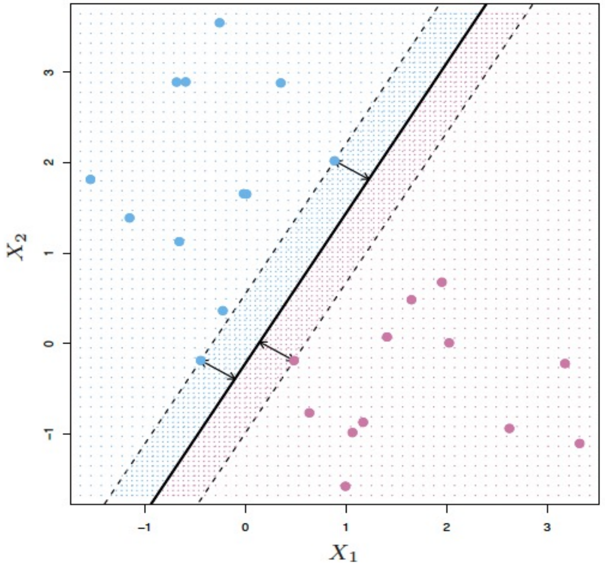
SVM and Generation of ML Algorithms

- Pre 1980:
  - Almost all learning methods learned linear decision surfaces. – Linear learning methods have nice theoretical properties
- 1980’s
  - Decision trees and NNs allowed efficient learning of non- linear decision surfaces
  - Little theoretical basis and all suffer from local minima
- 1990’s
  - Efficient learning algorithms for non-linear functions based on computational learning theory developed
  - Nice theoretical properties.
- 2010’s
  - Deep Neural Nets allow extremely efficient learning of non- linear decision surfaces
  - Little theoretical basis and all suffer from local minima

Support Vector Machine

- Constructs a *maximum margin* separator: a decision boundary with the largest possible distance to data points
  - This separator is linear and is also called hyperplane
    - $W \cdot X + b = 0$
  - Margin can be considered as a width of the *street* separating positive and negative training data points

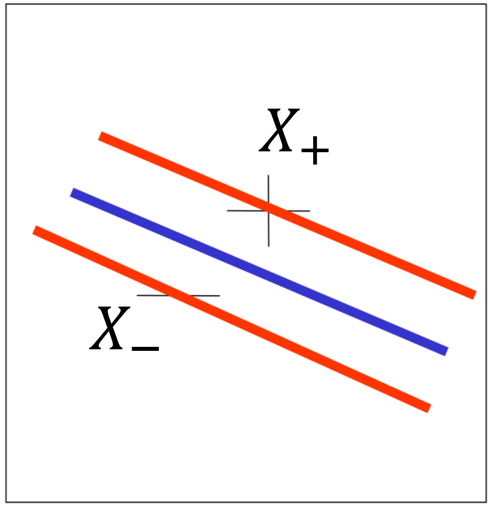




Notation:
X_1, X_2: features of the data

Support Vector Machine: Notaion

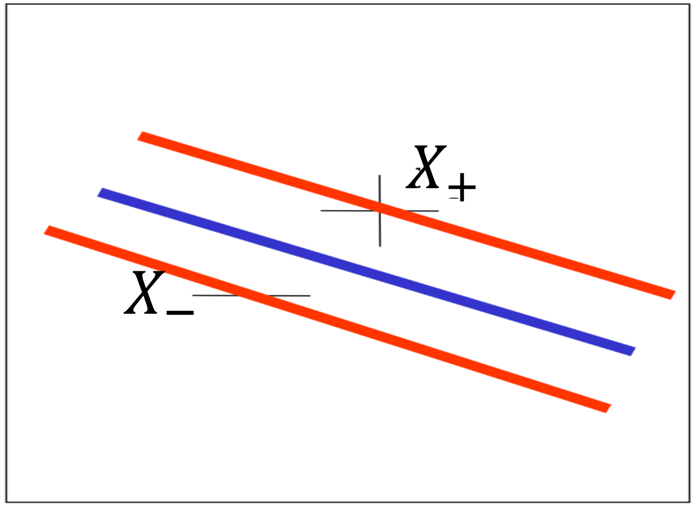
- Notation in SVM:
  - Class labels are +1 and -1 instead of +1 and 0
  - Intercept as a separate parameter: b
- W: vector perpendicular to the separator (blue line)
- For all positive data points or vectors:
  - $f(X_+) = W \cdot X_+ + b \geq +1$
- For all negative vectors:
  - $f(X_-) = W \cdot X_- + b \leq -1$
- Simply for all observations: $y_i(W \cdot X_i + b) \geq 1$



Notation: X_i: the data point

What is the margin?

- Let X_+ and X_- are the datapoints *closest* to separator
  - $W \cdot X_+ + b = +1$
  - $W \cdot X_- + b = -1$
  - Subtracting: $W \cdot (X_+ - X_-) = 2$



- Dividing by the length of W produces the distance between the lines:
  - $\frac{W}{||W||}(X_+ - X_-) = \frac{2}{||W||}$

- We want to maximize this distance

SVM: Honoring the Constraints

- Maximize: $\frac{2}{||W||}$
  - Equivalently, minimize: $||W||$ or $\frac{||W||^2}{2}$
    - Why this form?
      - For mathematical convenience

- Constraints: $y_i(W \cdot X_i + b) \geq 1$

- How do you solve it?
  - Lagrange method

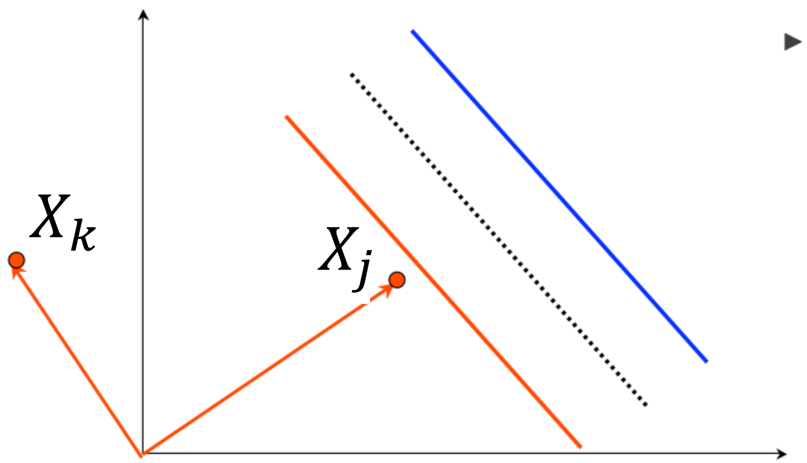
SVM: Optimization Problem

- Using LaGrange's method with α_i as Lagrange parameter for each observation, we have a quadratic optimization problem:
  - $argmax_{\alpha} \sum_j \alpha_j - \frac{1}{2} \sum_{j,k} \alpha_j \alpha_k y_j y_k (X_j \cdot X_k)$
    - Subject to: $\alpha_j \geq 0$ for all j and $\sum_j \alpha_j \, y_j = 0$

- Maximized if α_j's that correspond to the *support vectors (observations close to the separating hyperplane)* are non-zero
  - Those that 'matter' in fixing the maximum margin

Case 1: Two Completely Dissimilar Vectors

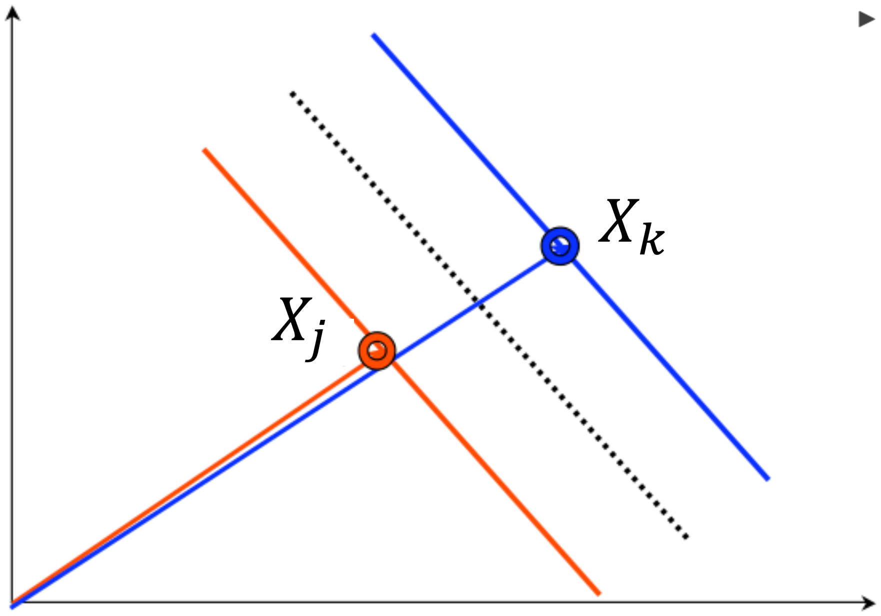
- 2 dissimilar (orthogonal) vectors: X_j, X_k, don't count at all



- $argmax_\alpha \sum_j \alpha_j - \frac{1}{2}\sum_{j,k} \alpha_j \alpha_k y_j y_k (X_j \cdot X_k)$
  - Subject to: $\alpha_j \geq 0$ for all j and $\sum_j \alpha_j \, y_j = 0$

Case 2.1: Two Alike Vectors from Different Class

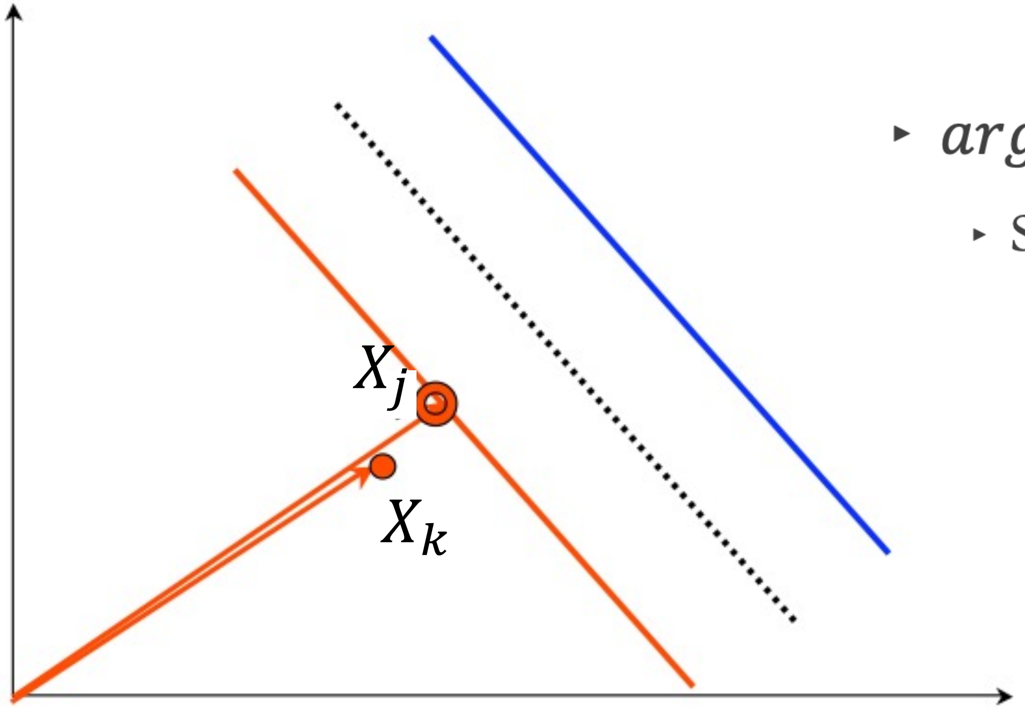
- 2 very similar X_j, X_k vectors that predict different classes tend to maximize the margin width



- $argmax_{\alpha} \sum_j \alpha_j - \frac{1}{2} \sum_{j,k} \alpha_j \alpha_k y_j y_k (X_j \cdot X_k)$
  - Subject to: $\alpha_j \geq 0$ for all j and $\sum_j \alpha_j y_j = 0$

Case 2: Two Alike Vectors from Same Class

- 2 vectors X_j, X_k that are similar but predict the same class are redundant



- $argmax_\alpha \sum_j \alpha_j - \frac{1}{2}\sum_{j,k} \alpha_j \alpha_k y_j y_k (X_j \cdot X_k)$
  - Subject to: $\alpha_j \geq 0$ for all j and $\sum_j \alpha_j y_j = 0$

SVM: Prediction

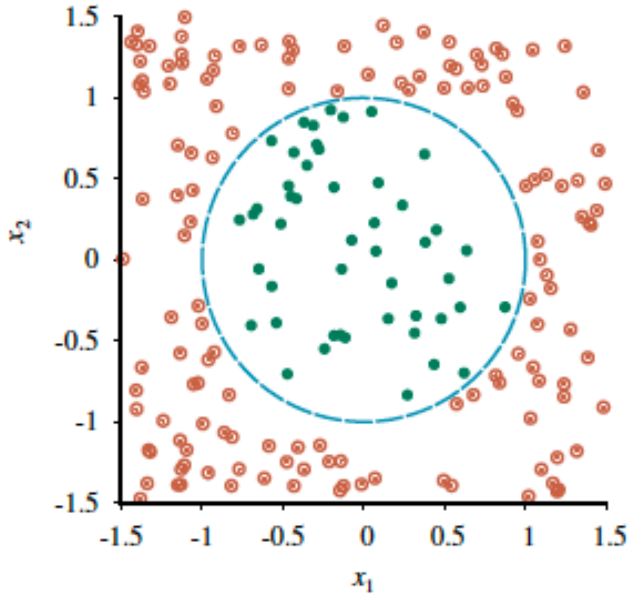
- $W = \sum_j \alpha_j \, y_j X_j$

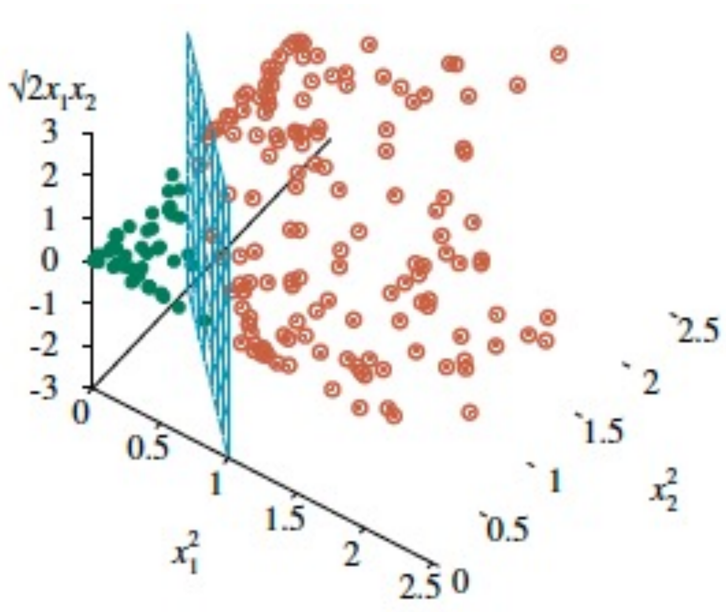
- For a test vector X:
  - $h(X) = sign(\sum_j \alpha_j \, y_j (X \cdot X_j) - b)$

What if the data is non-linearly separable?

- Transform the data into a different space
  - Gain linear separation by mapping the data to a higher dimensional space

- Ex: Data can be separated by a quadratic transformation

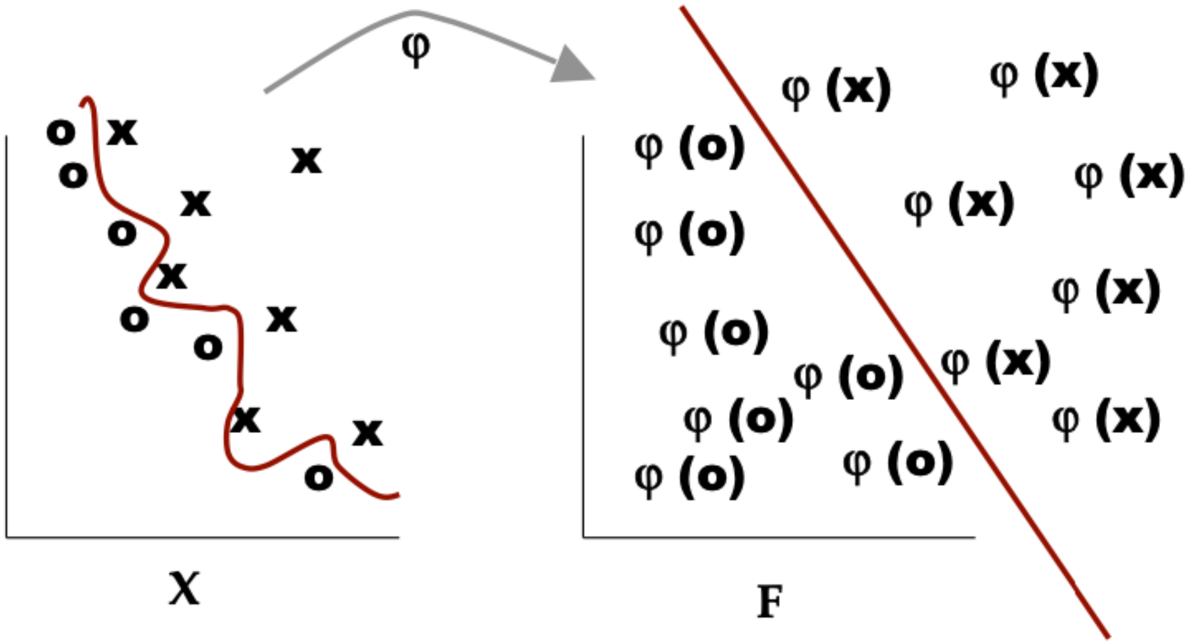




Notation: x_1, x_2 are features here

Objective Function in Transformed Space

- What we have:



- $argmax_\alpha \sum_j \alpha_j - \frac{1}{2} \sum_{j,k} \alpha_j \alpha_k y_j y_k (\varphi(X_j) \cdot \varphi(X_k))$
  - Subject to: $\alpha_j \geq 0$ for all j   and $\sum_j \alpha_j \, y_j = 0$

Objective Function in Transformed Space

- $argmax_\alpha \sum_j \alpha_j - \frac{1}{2}\sum_{j,k} \alpha_j \alpha_k y_j y_k (\varphi(X_j) \cdot \varphi(X_k))$
  - Subject to: $\alpha_j \geq 0$ for all j and $\sum_j \alpha_j \, y_j = 0$
- Simply, we can define a kernel function $K\left(X_j, X_k\right) = \varphi(X_j) \cdot \varphi(X_k)$
- Generalize the form of kernel as some function computes similarity in the transformed space
  - Polynomial kernel: $K\left(X_j, X_k\right) = (1 + X_j \cdot X_k)^d$
  - Radial Basis Function: $K\left(X_j, X_k\right) = e^{-\gamma |X_j - X_k|^2}$
- SVM with non-linear kernels are helpful when the data is too noisy

Evaluation Metrics for the Binary Classifier

- Binary Classification Problem:

- Counts of:
  - True Positive (TP)
  - False Positive (FP)
  - True Negative (TN)
  - False Negative (FN)

		Truth	
		True	False
Prediction	True	TP	FP
	False	FN	TN

- Accuracy: $\frac{\text{TP+TN}}{\text{TP+TN+FP+FN}}$

Sensitivity and Specificity

- Sensitivity: $\dfrac{\text{TP}}{\text{TP+FN}}$

- Specificity: $\dfrac{\text{TN}}{\text{TN+FP}}$

		Truth	
		True	False
Prediction	True	TP	FP
	False	FN	TN

Precision & Recall

- Precision (P): $\frac{\text{TP}}{\text{TP+FP}}$
- Recall (R): $\frac{\text{TP}}{\text{TP+FN}}$ (same as sensitivity)
- F1-Measure: $\frac{2*\text{P}*\text{R}}{\text{P+R}}$
  - F1-measure is used when we have imbalance data

		Truth	
		True	False
Prediction	True	TP	FP
	False	FN	TN

Performance of Logistic Regression and SVM

```
print (classification_report(y_test  , lr.predict(x_test)))
```

```
              precision    recall  f1-score   support

           0       0.86      0.96      0.91       367
           1       0.58      0.25      0.35        77

    accuracy                           0.84       444
   macro avg       0.72      0.60      0.63       444
weighted avg       0.81      0.84      0.81       444
```

```
print (classification_report(y_test  , svc.predict(x_test)))
```

```
Training Accuracy: 0.8600451467268623
Test Accuracy: 0.831081081081081
              precision    recall  f1-score   support

           0       0.84      0.98      0.91       367
           1       0.56      0.12      0.19        77

    accuracy                           0.83       444
   macro avg       0.70      0.55      0.55       444
weighted avg       0.79      0.83      0.78       444
```

- Performing poorly on class 1
  - Poor in identifying customers who will respond
  - One way: Change the thresholds of classifier

What is Optimal Threshold for Classification?

- Average response rate in the test data: 0.17
- The default cut-off values such as 0.5, does not work given the low base rate of responses
- We need to find a better threshold
  - Lift value
  - Youden's index

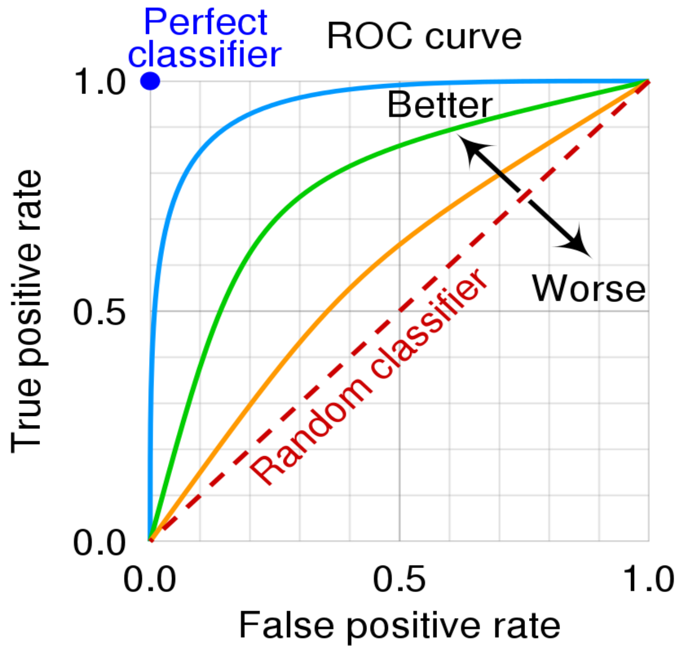
Optimal Threshold for Classification: Lift Value

- Lift calculates the response rate that predictive model provides over the average response rate in the data
- We calculate deciles based on the probability of responding that is ordered from highest to lowest
- Calculate the lift for each decile
  - Ratio of response rate to the average response rate, for each decile
- Use the probability value corresponding to lift value of 2 as the threshold
  - Lift value of 2 says that these customers are twice as likely to respond compared to the average customer in the data

Optimal Threshold for Classification: Youden's Index

- Receiver Operating Characteristics (ROC) curve gives the classification performance at all thresholds
    - True positive rate: sensitivity
    - False positive rate: 1- specificity

- Identify the threshold at which the ROC curve is farthest from the random classifier performance
  - Youden's index or Youden's J statistic
    - Calculated as: true positive rate – false positive rate
  - Optimal Threshold: Threshold at which Youden's index is *maximum*



= 1-specificity

Classification with Different Thresholds

- Lift value:

```
#threshold corresponding to lift value of 2
threshold = 0.22
preds = np.where(y_pred2[:,1] > threshold, 1, 0)
print(classification_report(y_test  , preds))
```

	precision	recall	f1-score	support
0	0.90	0.85	0.88	367
1	0.45	0.57	0.50	77
accuracy			0.80	444
macro avg	0.68	0.71	0.69	444
weighted avg	0.83	0.80	0.81	444

```
#threshold corresponding to lift value of 2
threshold = 0.31
preds = np.where(y_svc_pred2[:,1] > threshold, 1, 0)
print(classification_report(y_test  , preds))
```

	precision	recall	f1-score	support
0	0.85	0.97	0.91	367
1	0.57	0.21	0.30	77
accuracy			0.84	444
macro avg	0.71	0.59	0.61	444
weighted avg	0.80	0.84	0.80	444

Logistic Regression

SVM with linear kernel

- Youden's index:

```
0.1632311427133435
```

Use the threshold based on Youden's index and obtain classification report.

```
#threshold corresponding to lift value of 2
threshold = optimal_threshold
preds = np.where(y_pred2[:,1] > threshold, 1, 0)
print(classification_report(y_test  , preds))
```

	precision	recall	f1-score	support
0	0.93	0.79	0.85	367
1	0.42	0.73	0.53	77
accuracy			0.78	444
macro avg	0.68	0.76	0.69	444
weighted avg	0.84	0.78	0.80	444

```
0.12030979656781648
```

```
#threshold corresponding to lift value of 2
threshold = optimal_threshold
preds = np.where(y_svc_pred2[:,1] > threshold, 1, 0)
print(classification_report(y_test  , preds))
```

	precision	recall	f1-score	support
0	0.93	0.68	0.79	367
1	0.33	0.75	0.46	77
accuracy			0.69	444
macro avg	0.63	0.72	0.62	444
weighted avg	0.83	0.69	0.73	444

Thank You