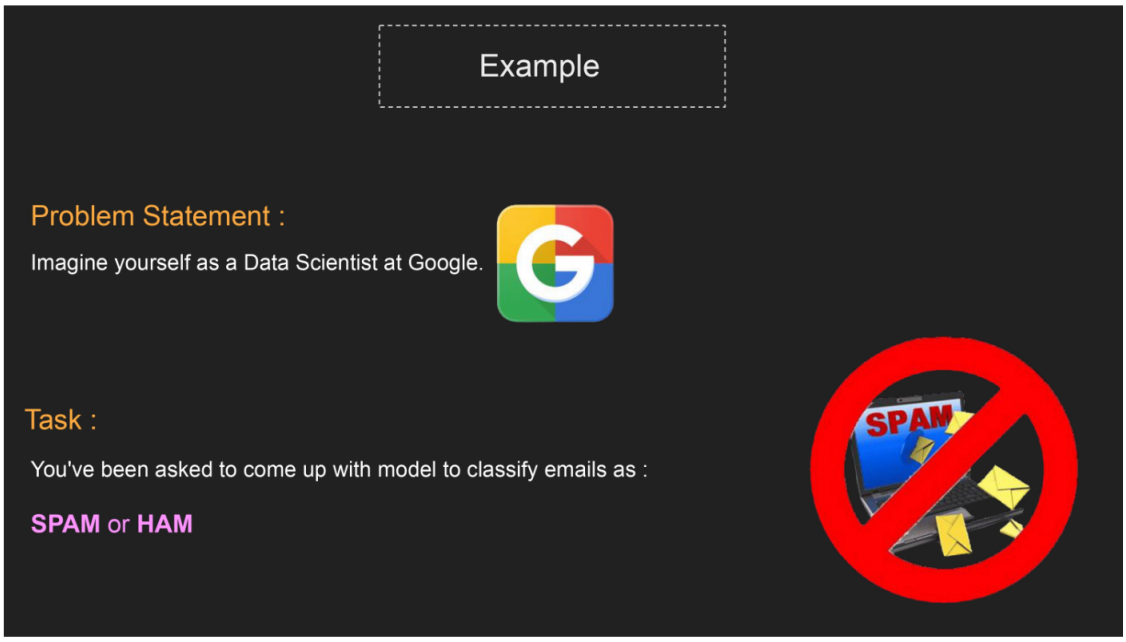
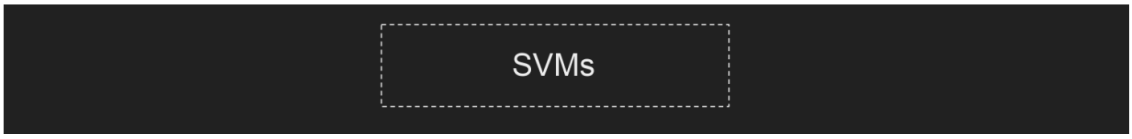
Content

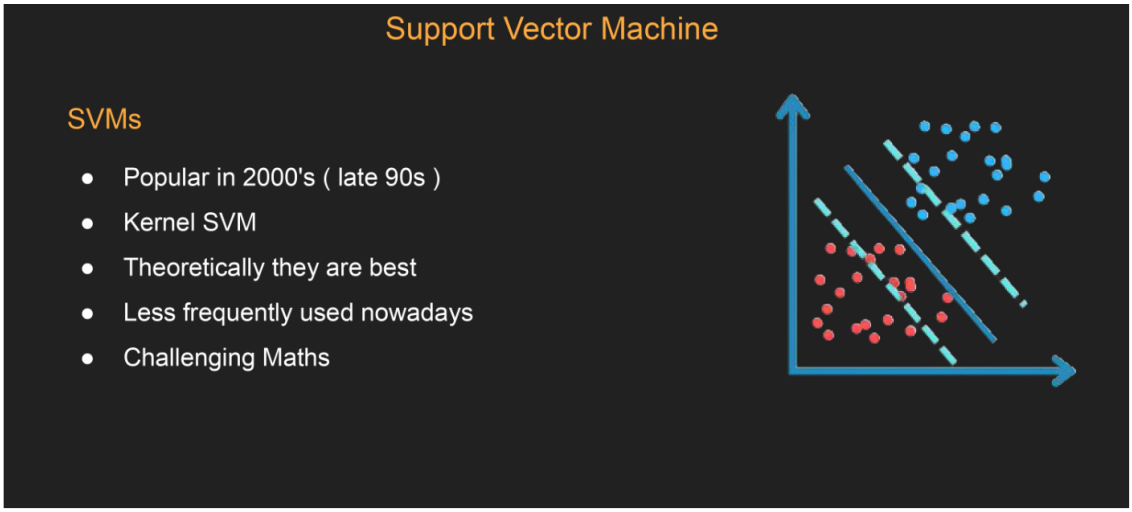
- Problem Statement
- Geometric Intuition of SVM
- Hard Margin SVM
- Soft Margin SVM
- Algebric Intuition of SVM
- Intution of Hinge Loss
- SVM Imbalance
- Code Implementation of Linear SVM

Problem Statement

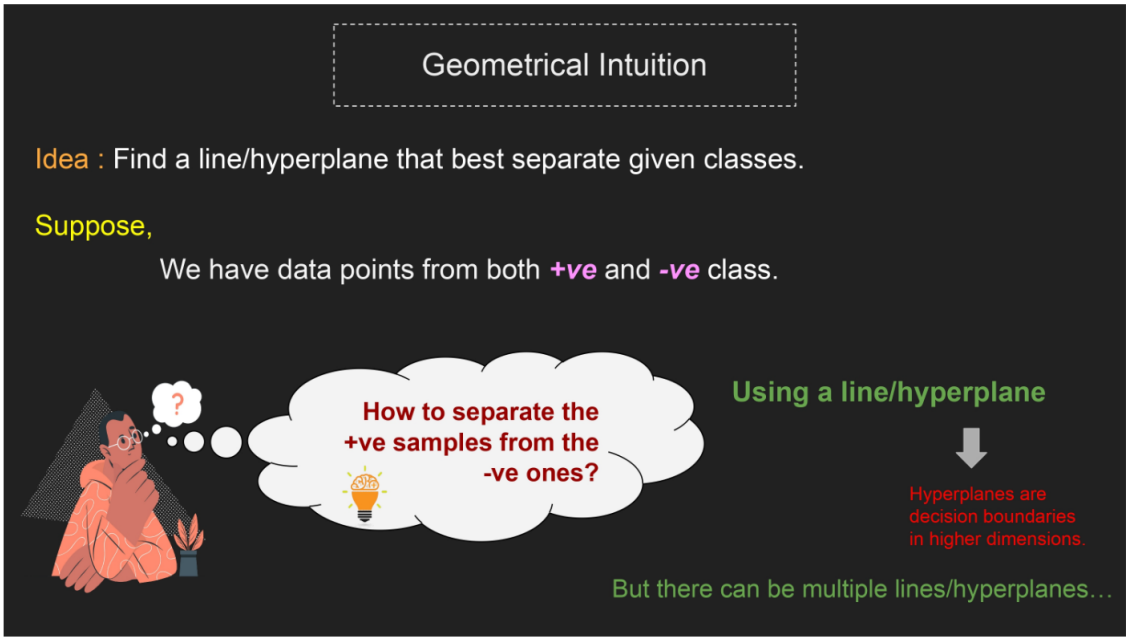


SVM - Support Vector Machine





Geometric intution behind SVM



Why do we need a large margin?

Because,

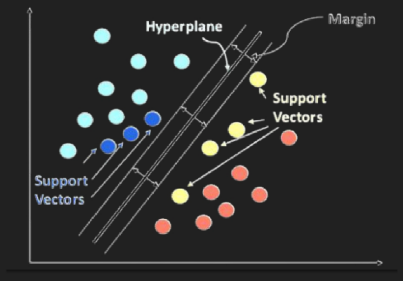
Larger the margin,

Better the separation

between the *+ve* and *-ve* samples.

Such classifiers where,

The margin should be as large as possible
are called **margin-maximizing classifiers.**

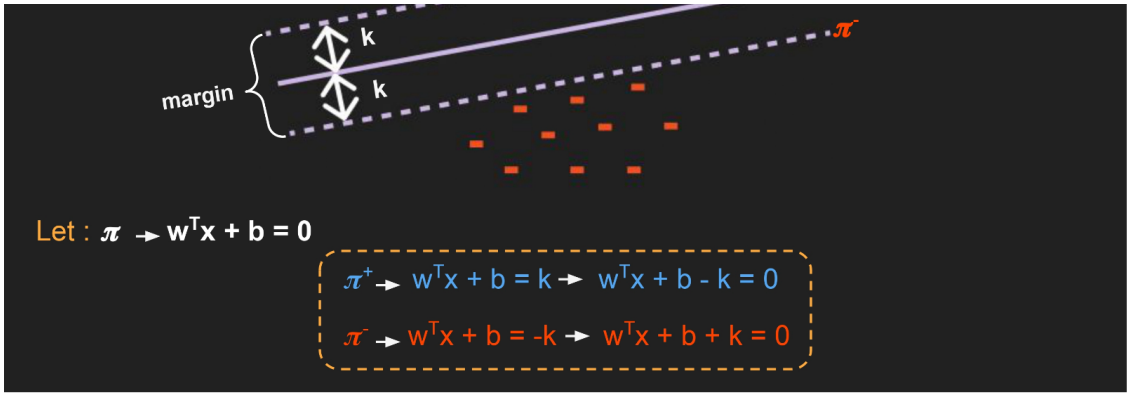


NOTE: Distances from data points are measured perpendicular to any of the hyperplanes.

Assume we have :

- +ve and -ve data points
- π is a margin-maximizing hyperplane
- π^+ is the +ve hyperplane parallel to
- π^+ is touching the closest +ve points to π
- π^- is the -ve hyperplane parallel to
- π^- is touching the closest -ve points to π
- Margin is the dist. between π+ and π-

How do we define the hyperplanes?



Let : $\pi \rightarrow w^T x + b = 0$

$\pi^+ \rightarrow w^T x + b = k \rightarrow w^T x + b - k = 0$

$\pi^- \rightarrow w^T x + b = -k \rightarrow w^T x + b + k = 0$

What will be the length of margin?

According to linear algebra, distance from origin :-

For : $\pi^+ : d(0, \pi^+) = \frac{b - k}{||w||}$

For : $\pi^- : d(0, \pi^-) = \frac{b + k}{||w||}$

So,

$$margin\ d(\pi^+, \pi^-) = d(0, \pi^-) - d(0, \pi^+) = \frac{2k}{||w||}$$

What are the parameters of the margin?

- weight **(w)**
- constant **(b)**

Consider these two cases :

Case 1

k = 1

$$d = \frac{2 * 1}{||w||} = \frac{2}{||w||}$$

$$max_{w,b}(\frac{2}{||w||})$$

Case 2

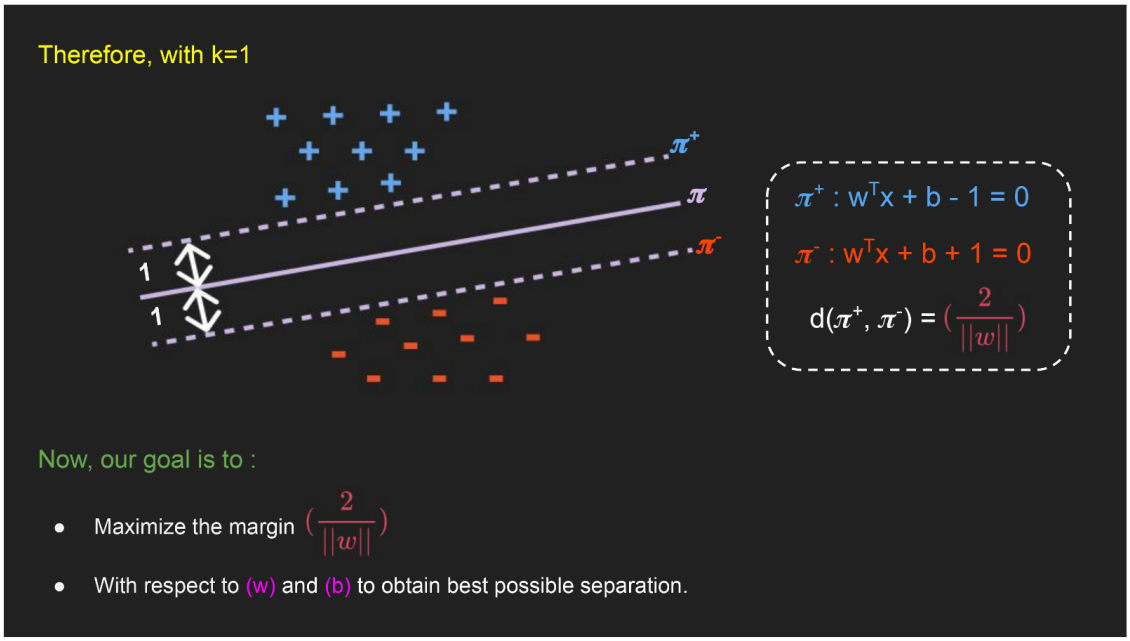
k = 10

$$d = \frac{2 * 10}{||w||} = \frac{20}{||w||}$$

$$max_{w,b}(\frac{20}{||w||})$$

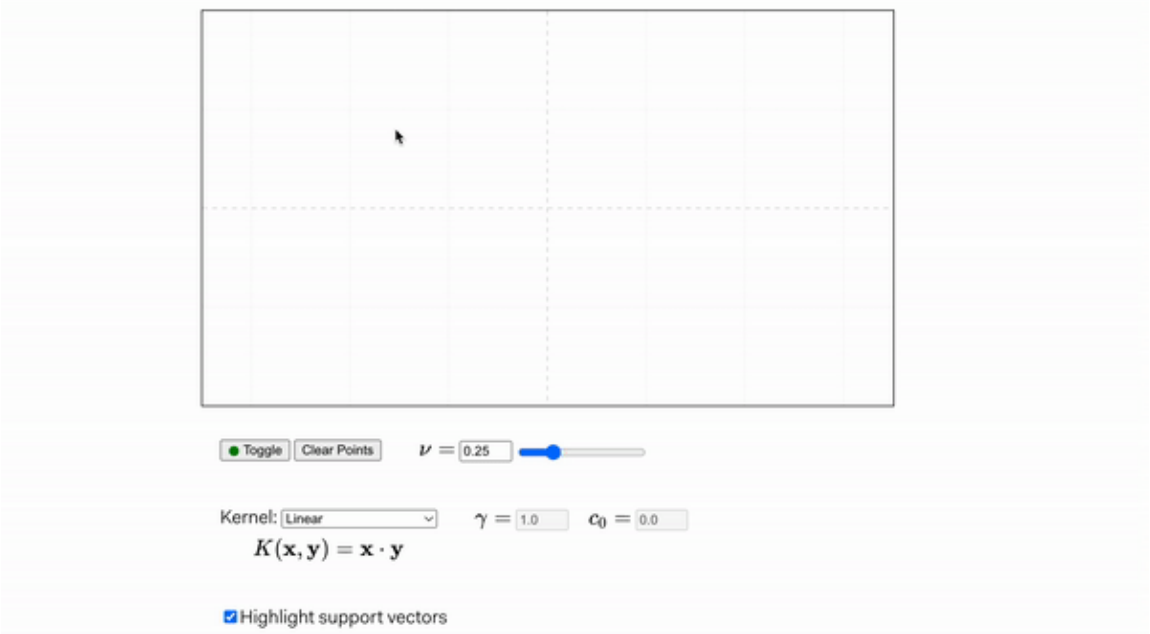
Changing the value of (k) would only scale the margin but won't affect the position of the hyperplane.

Hence, for mathematical simplicity we take k=1

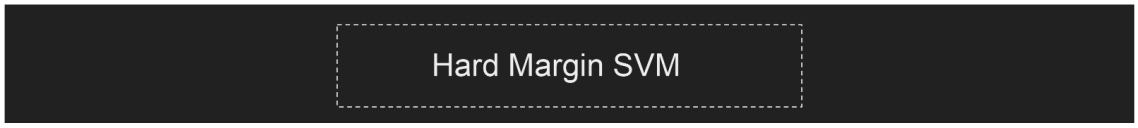


SVM Demo

https://jgreitemann.github.io/svm-demo



Hard Margin SVM



In a Binary Classification problem,

y_i = +1 for +ve samples

y_i = -1 for -ve samples

Now the **functional margin** i.e. the signed distance between

- the data point x_i and
- the hyperplane π is…

  $= y_i(w^T x + b)$

Hard Margin SVM

Our goal is to maximize the margin
s.t. the +ve and -ve samples are on different sides of hyperplane π

Therefore, the margin constraints are :

For $y_i = +1 \longrightarrow \pi^+ = y_i (w^T x + b) \geq 1$

For $y_i = -1 \longrightarrow \pi^- = y_i (w^T x + b) \leq -1$

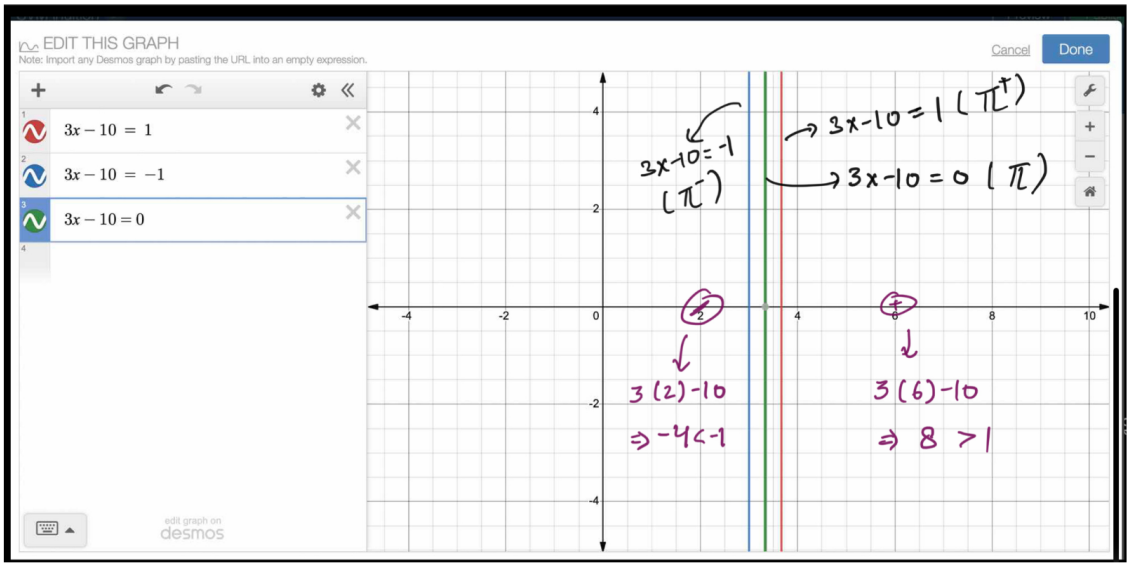
If we club both these equations:

$$argmax_{w,b}\left(\frac{2}{||w||}\right) \; s.t. \; y_i(w^T x_i + b) \geq 1$$
$$\forall \; i : 1 \longrightarrow N$$

How does $y_i\ (w^Tx + b) \geq 1$ works?

For +ve samples	For -ve samples
y_i = +1 (+ve)	y_i = -1 (-ve)
$w^Tx_i + b \geq 1$ (+ve)	$w^Tx_i + b \leq -1$ (-ve)
+ve * +ve = +ve	-ve * -ve = +ve

Example -



Why the $y_i\ (w^Tx + b) \geq 1$ constraint?

Our goal is to:

- completely seperate the two classes
- with a margin as large as possible

With this constraint:

- all +ve data points should lie beyond π^+
- all -ve data points should lie beyond π^-

This is called **Hard Margin Classification**

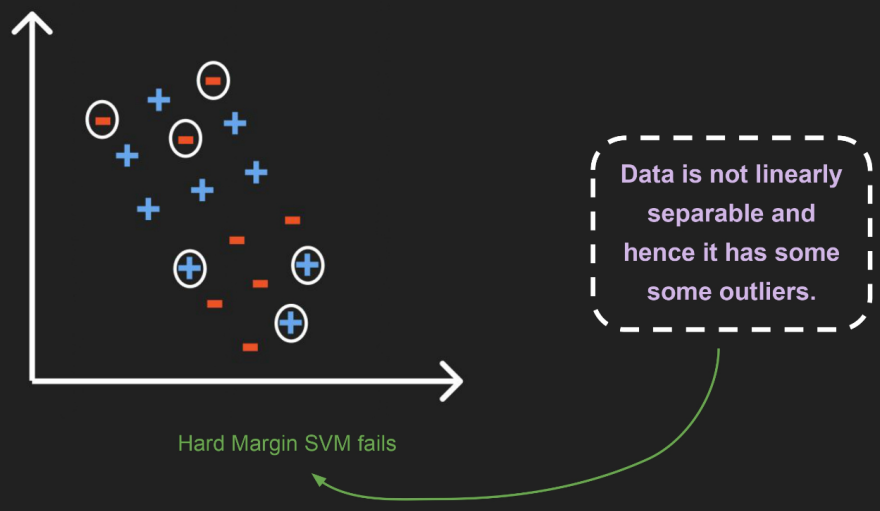
Which means, we strictly impose that:

All instances must be

- off the margin and

- on the correct side
- i.e., Zero Error

When would a linear SVM with hard margin fail?

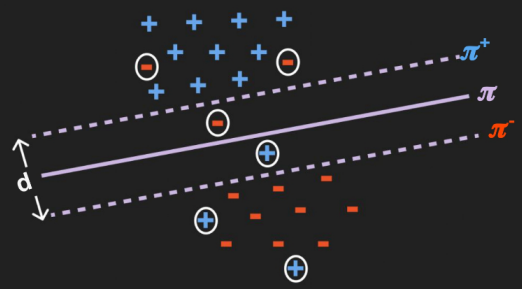


Soft Margin SVM

Soft Margin SVM

What if the data isn't perfectly linearly separable?

Some data points lie on the wrong side of the hyperplane π.



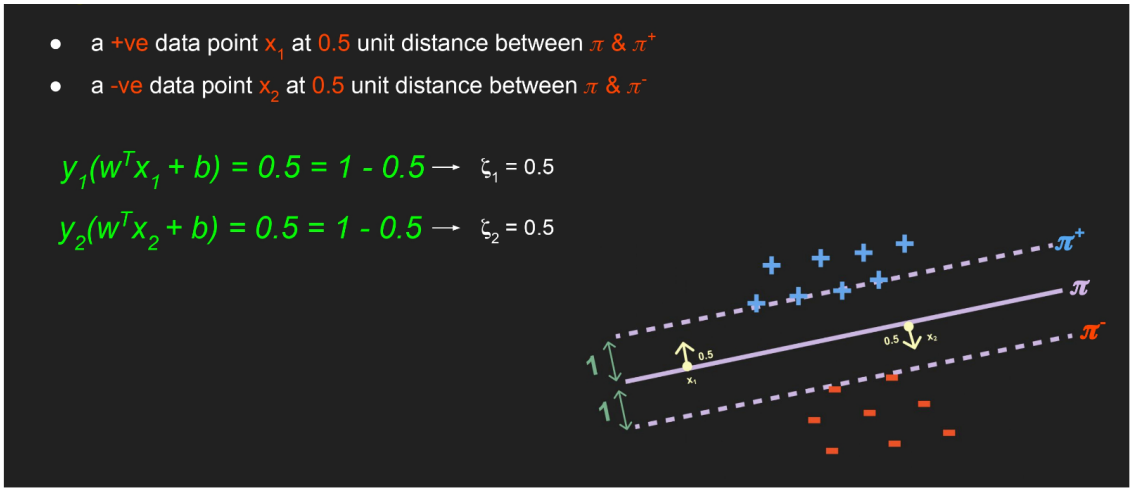
Almost linearly separable

Our goal is to :

- Maximize the margin
- Minimize data points with error $\zeta_i > 0$
- Minimize the errors ζ_i 's

How to account for these data points?

Imagine,



Now, our optimization problem becomes:

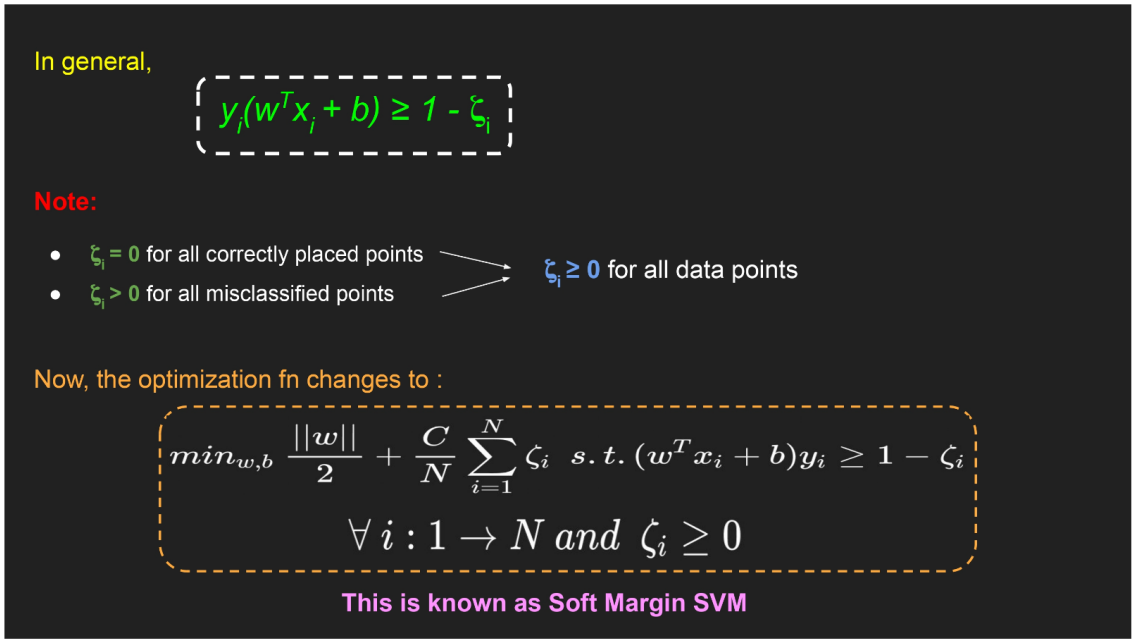
- $max\ \frac{2}{||w||}$ i.e., the margin
- along with minimizing error $\zeta_i's$

because we're try to get the best possible classificaton.

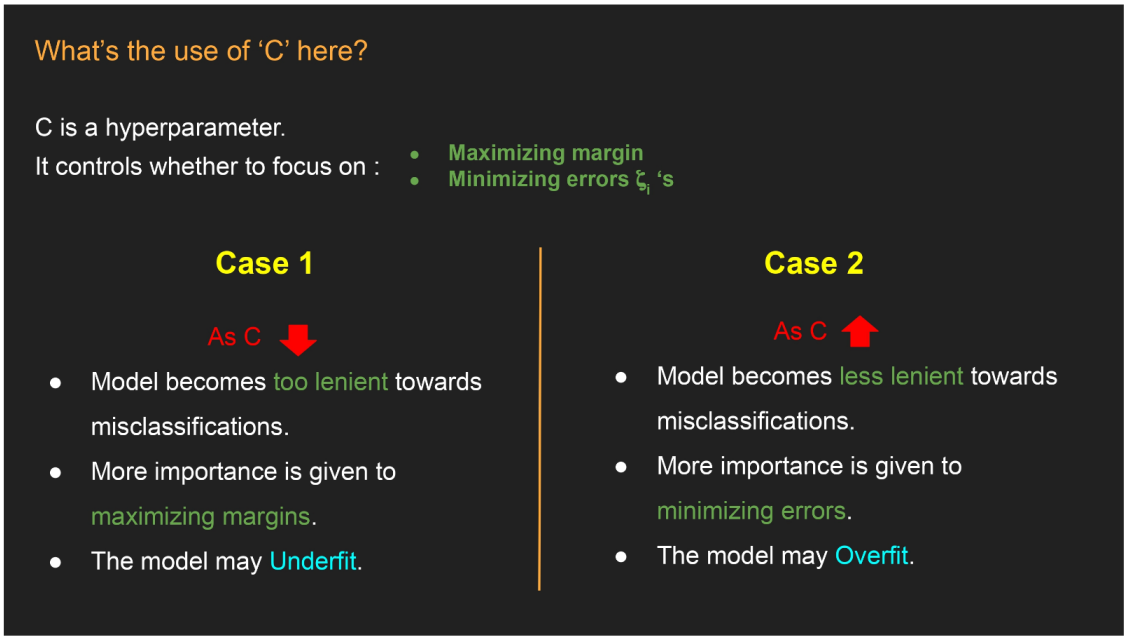
Can we think of another way to write this?

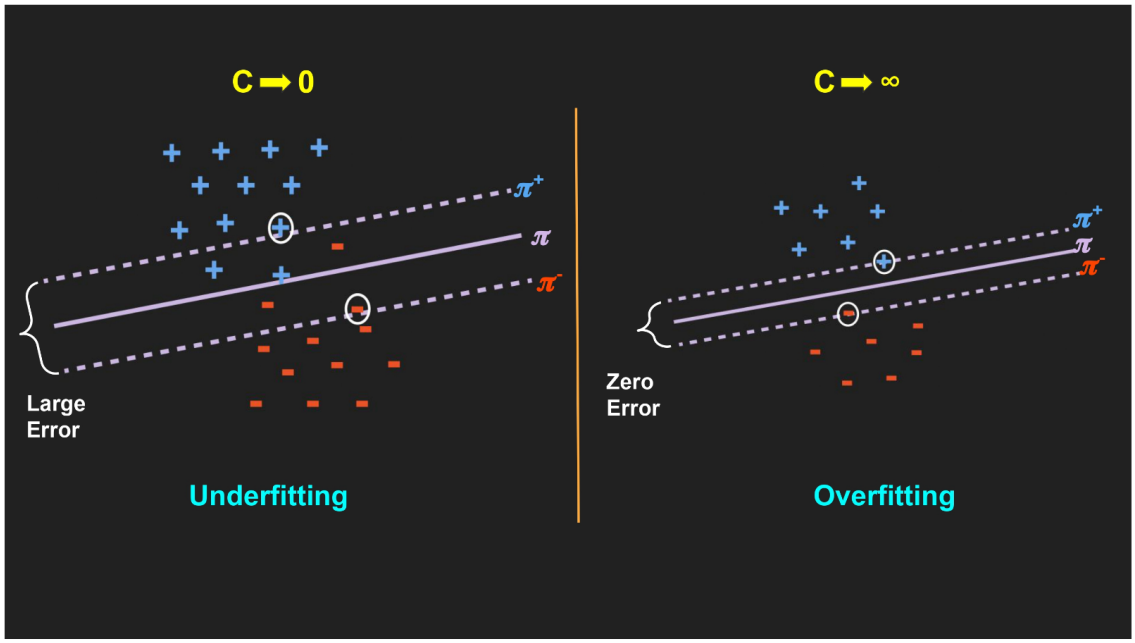
Reciprocating above equation,

- $min\ \frac{||w||}{2}$ with $\zeta_i's$



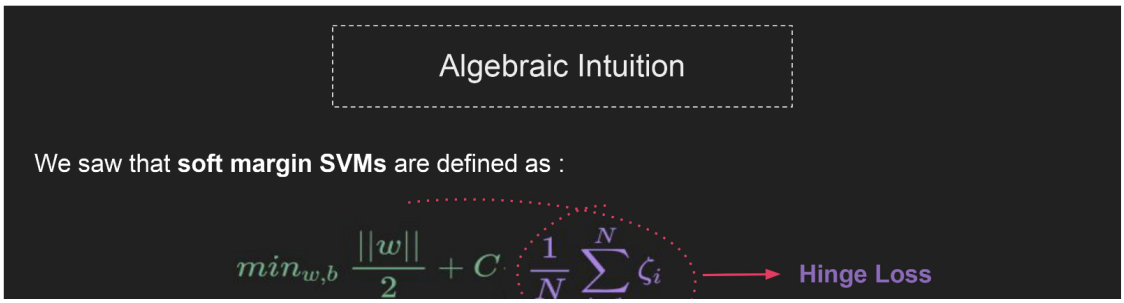
Hyperparameters in SVM

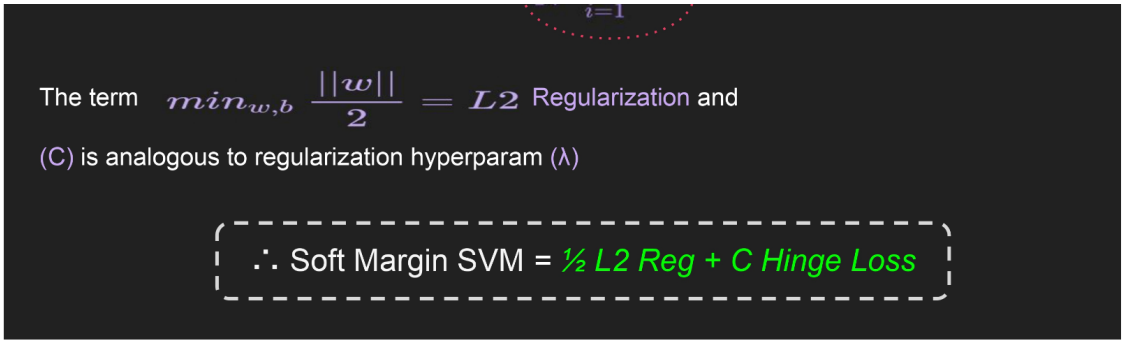




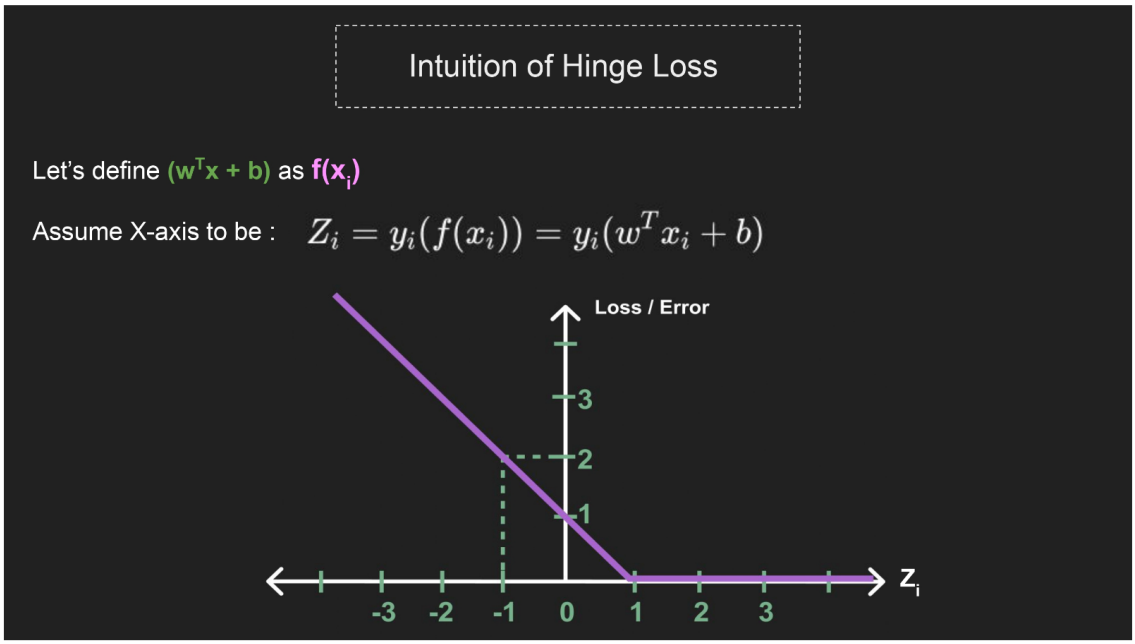
Therefore, we need to find a balance here.

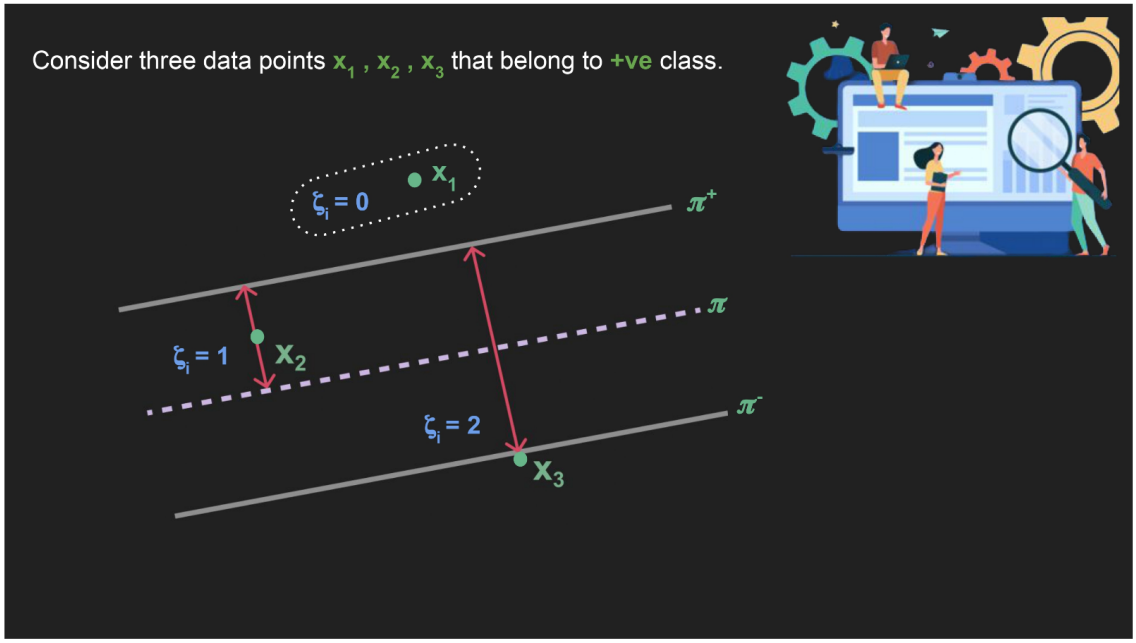
Algebraic intuition behind SVM




The term $min_{w,b} \frac{||w||}{2} = L2$ Regularization and
(C) is analogous to regularization hyperparam (λ)
∴ Soft Margin SVM = ½ L2 Reg + C Hinge Loss

Hinge Loss


Intuition of Hinge Loss
Let's define $(w^T x + b)$ as $f(x_i)$
Assume X-axis to be : $Z_i = y_i(f(x_i)) = y_i(w^T x_i + b)$
Loss / Error
3
2
1
-3 -2 -1 0 1 2 3
Z_i


Consider three data points x_1 , x_2 , x_3 that belong to +ve class.
$\xi_i = 0$
x_1
π^+
π
$\xi_i = 1$
x_2
$\xi_i = 2$
π^-
x_3

Case 1	Case 2	Case 3
• Point x_1 , lies on/beyond π^+	• Point x_2 , lies on π	• Point x_3 , lies on π^-
• Then $Z_i \geq 1$	• Then $Z_i = 0$	• Then $Z_i = -1$
$Y_i (w^T x + b) \geq 1$	$Y_i (w^T x + b) = 0$	$Y_i (w^T x + b) = -1$
• So, $\zeta_i = 0$	• So, $\zeta_i = 1$	• So, $\zeta_i = 2$
$d (x_1 , \pi^+) = 0$	$d (x_2 , \pi^+) = 1$	$d (x_3 , \pi^+) = 2$

What can we conclude from this?

- Error cannot be negative. $\zeta_i \geq 0$
- According to the constraint -

$y_i (w^T x + b) > - 1 - \zeta$

$= Z_i > 1 - \zeta$

$= \zeta_i \geq 1 - \zeta - Z_i$

We can combine these two equations and get Hinge Loss $\zeta_i = max (0 , 1 - Z_i)$

NOTE: As Z_i ⬆ ζ_i ⬇ till zero

As Z_i ⬇ ζ_i ⬆

In [ ]:

Comparison with Log Loss

Comparison with Log Loss

Logistic Regression	Support Vector Machine
• $y_i \in \{ 0 , 1 \}$	• $y_i \in \{ -1 , 1 \}$

- So, Log Loss =

$$\sum_{i=1}^{N} y_i log(\hat{y_i}) + (1 - y_i) \log(1 - \hat{y_i})$$

- So, Hinge loss = $\sum_{i=1}^{N} \zeta_i$

$$\sum_{i=1}^{N} max(0, 1 - Z_i)$$

What happens if y ∈ { -1 , +1 } for Log Loss?

If y ∈ { -1 , +1 }
Then :

$$\text{Log Loss} = \sum_{i=1}^{n} log(1 + e^{(-z_i)})$$

$$= \sum_{i=1}^{n} log(1 + e^{(-y_i(w^T x_i + b))})$$

We will not be deriving how we get this equation.

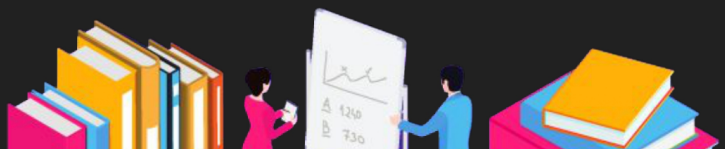
Data Imbalance

Are SVMs affected by class imbalance?

Only a few data points contribute to the **Hinge loss (ζ_i)** .
These points are called **Support Vectors**.

Hence, SVM is only affected by **imbalance in no. of support vectors** for each class.

NOTE: The balance in no. of support vectors can't be guaranteed.



Code implementation of Linear SVM

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from collections import Counter
from sklearn import feature_extraction, model_selection, naive_bayes, metric
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.preprocessing import StandardScaler
import warnings

warnings.filterwarnings('ignore')
%matplotlib inline
```

In [ ]:
```
!gdown 1QViUZJ5UIBCgxB_qbOXTLs_2V48w7MWo

df = pd.read_csv('Spam_processed.csv', encoding='latin-1')
df.dropna(inplace = True)
```

```
Downloading...
From: https://drive.google.com/uc?id=1QViUZJ5UIBCgxB_qbOXTLs_2V48w7MWo
To: /content/Spam_processed.csv

  0% 0.00/767k [00:00<?, ?B/s]
 68% 524k/767k [00:00<00:00, 4.28MB/s]
100% 767k/767k [00:00<00:00, 5.50MB/s]
```

In [ ]:
```
df
```

Out[ ]:

	type	message	cleaned_message
0	0	Go until jurong point, crazy.. Available only ...	go jurong point crazy available bugis n great ...
1	0	Ok lar... Joking wif u oni...	ok lar joking wif u oni
2	1	Free entry in 2 a wkly comp to win FA Cup fina...	free entry 2 wkly comp win fa cup final tkts 2...
3	0	U dun say so early hor... U c already then say...	u dun say early hor u c already say
4	0	Nah I don't think he goes to usf, he lives aro...	nah nt think goes usf lives around though
...	...	...	...
5567	1	This is the 2nd time we have tried 2 contact u...	2nd time tried 2 contact u u å750 pound prize ...
5568	0	Will Ì_ b going to esplanade fr home?	ì_ b going esplanade fr home
5569	0	Pity, * was in mood for that. So...any other s...	pity mood suggestions
5570	0	The guy did some bitching but I acted like i'd...	guy bitching acted like interested buying some...
5571	0	Rofl. Its true to its name	rofl true name

5565 rows × 3 columns

- Performing train-test split
- with CountVectorization
- and StandardScaler.

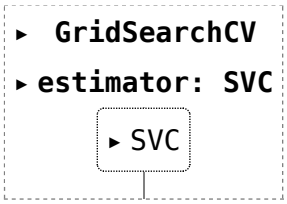
In [ ]:
```
from sklearn.model_selection import train_test_split

df_X_train, df_X_test, y_train, y_test = train_test_split(df['cleaned_messag
                                                          test_size=0.25, ra
print([np.shape(df_X_train), np.shape(df_X_test)])

# CountVectorizer
f = feature_extraction.text.CountVectorizer()
X_train = f.fit_transform(df_X_train)
X_test = f.transform(df_X_test)

# StandardScaler
scaler = StandardScaler(with_mean=False) # problems with dense matrix
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

print([np.shape(X_train), np.shape(X_test)])
print(type(X_train))
```

```
[(4173,), (1392,)]
[(4173, 7622), (1392, 7622)]
<class 'scipy.sparse._csr.csr_matrix'>
```

Let's train Linear SVM on the given Spam/Ham data.

In [ ]:
```
# SVC

from sklearn.svm import SVC

from sklearn.model_selection import GridSearchCV

params = {
          'C': [1e-4,  0.001, 0.01, 0.1, 1,10] # which hyperparam value of C
         }

svc = SVC(class_weight={ 0:0.1, 1:0.5 }, kernel='linear')
clf = GridSearchCV(svc, params, scoring = "f1", cv=3)

clf.fit(X_train, y_train)
```

Out[ ]:



In [ ]:
```
res = clf.cv_results_

for i in range(len(res["params"])):
  print(f"Parameters:{res['params'][i]} \n Mean score: {res['mean_test_score
```

```
Parameters:{'C': 0.0001}
 Mean score: 0.6566305780023073
 Rank: 6
Parameters:{'C': 0.001}
 Mean score: 0.7742322485787693
 Rank: 1
Parameters:{'C': 0.01}
 Mean score: 0.767533370474547
 Rank: 2
Parameters:{'C': 0.1}
 Mean score: 0.7649416969151316
 Rank: 3
Parameters:{'C': 1}
 Mean score: 0.7649416969151316
 Rank: 3
Parameters:{'C': 10}
 Mean score: 0.7649416969151316
 Rank: 3
```

As you can see,

- we get the best performance when $C = 0.001$,
- with F1 Score of 0.77.

Now implementing this SVM on the test data.

In [ ]:
```
svc = SVC(C=0.001,class_weight={ 0:0.1, 1:0.5 }, kernel='linear')

svc.fit(X_train, y_train)

y_pred = svc.predict(X_test)

print(metrics.f1_score(y_test,y_pred))
```

```
0.8835820895522388
```

Linear SVM performs much well

- on the Spam/Ham data
- with F1 Score of 0.88
- when using class weights.