Support Vector Machines

Speaker: Zhiqing Yang, Weiwen Min

Overview

1. What is SVM?

2. Key concepts of SVM

3. Mathematical principles of SVM

4. Implementation of SVM in Python

5. Advantages and Disadvantages

6. Development Trends of SVM

1. What is SVM?

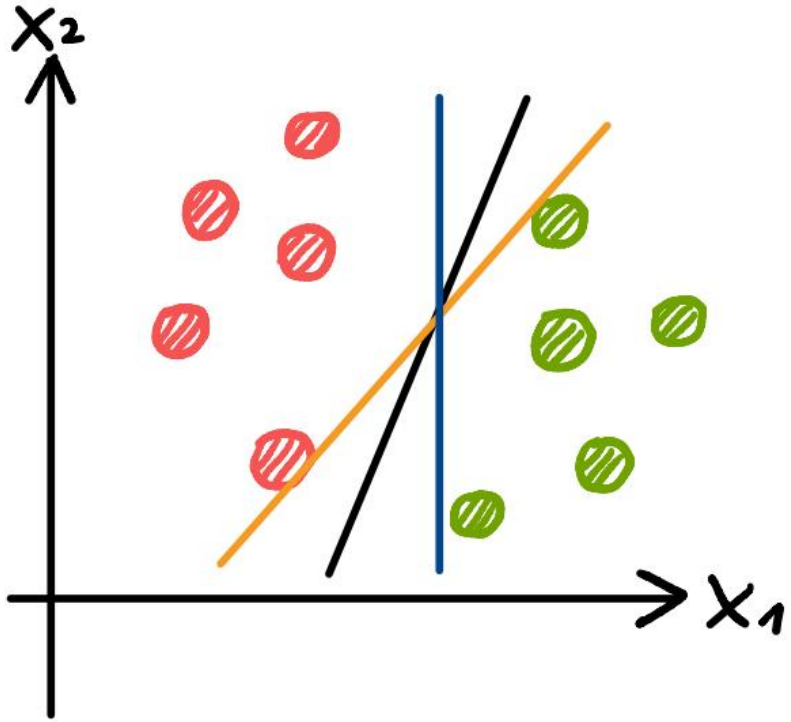
What is Support Vector Machine?

- Support vector machines (SVMs) are a set of **supervised** learning methods used for:
  - classification
  - regression
  - outliers detection

- It holds a significant place in machine learning:
  - Strong Theoretical Foundationregression
  - Effective in High-Dimensional Spaces
  - Versatility with Kernels
  - Margin Maximization
  - Widespread Applications
  - Interpretability

What is the goal of SVM?

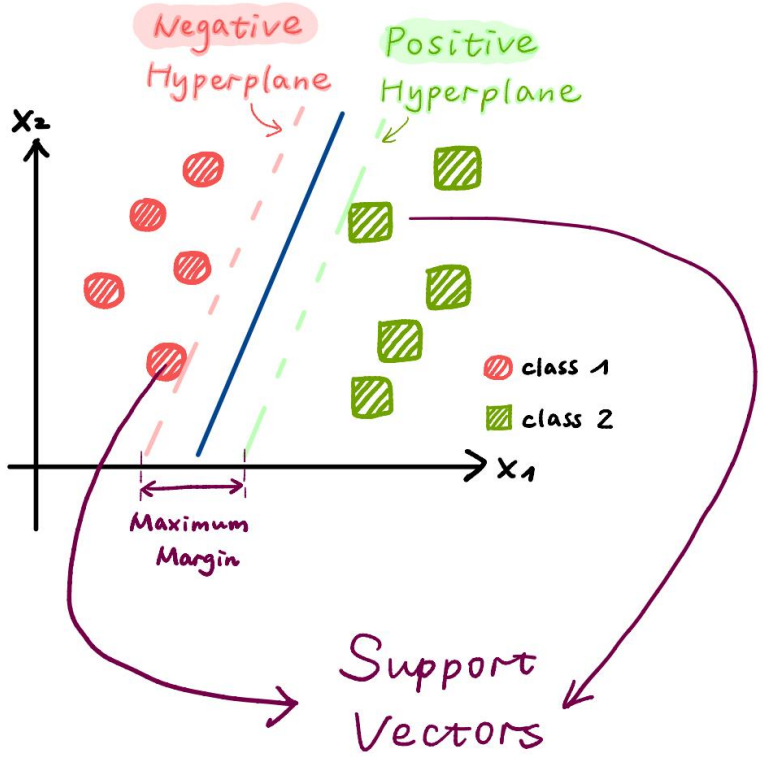
- to find a **hyperplane(decision boundary)** that best separates data points into two different classes



2. Key concepts of SVM

Key Concepts in SVM

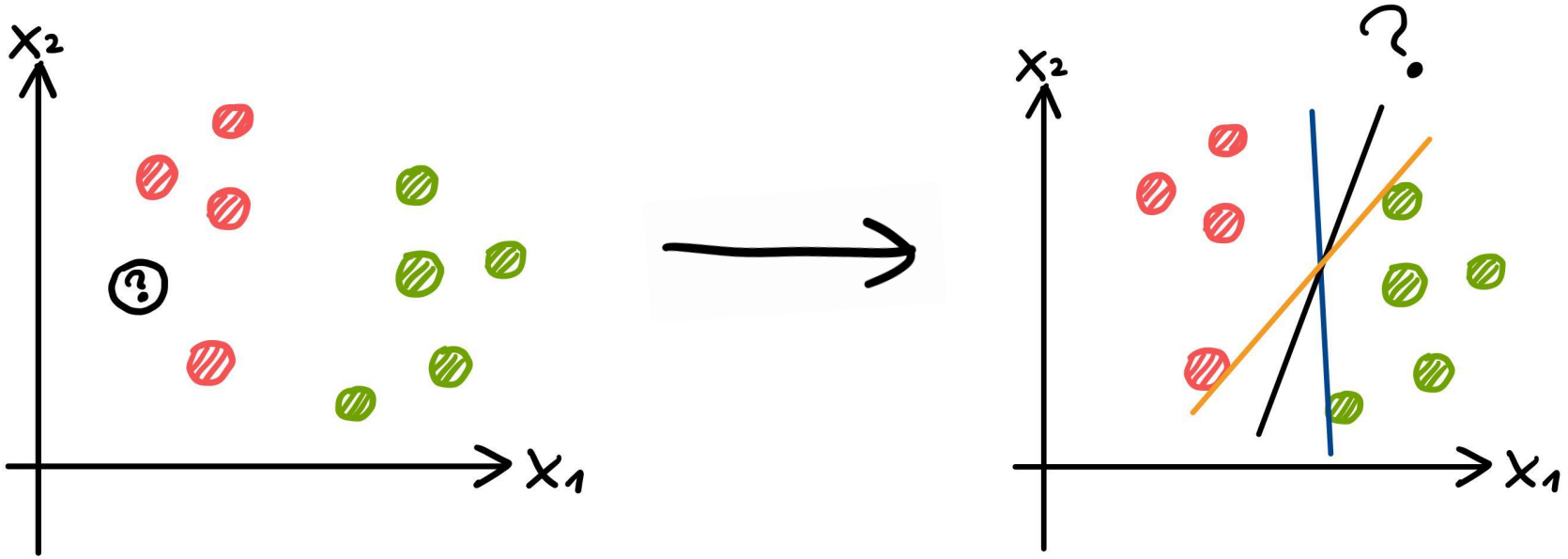
- **Support Vectors:** These are the data points closest to the hyperplane and highly influence its position and orientation.
- **Margin**: The margin is the distance between the hyperplane and the nearest data point from either class. A larger margin implies a better generalization of the classifier.
- **Hyperplane**: In a two-dimensional space, this is a line that separates the data into two parts. In higher dimensions, it's a plane or a higher-dimensional analog.



How does SVM work?

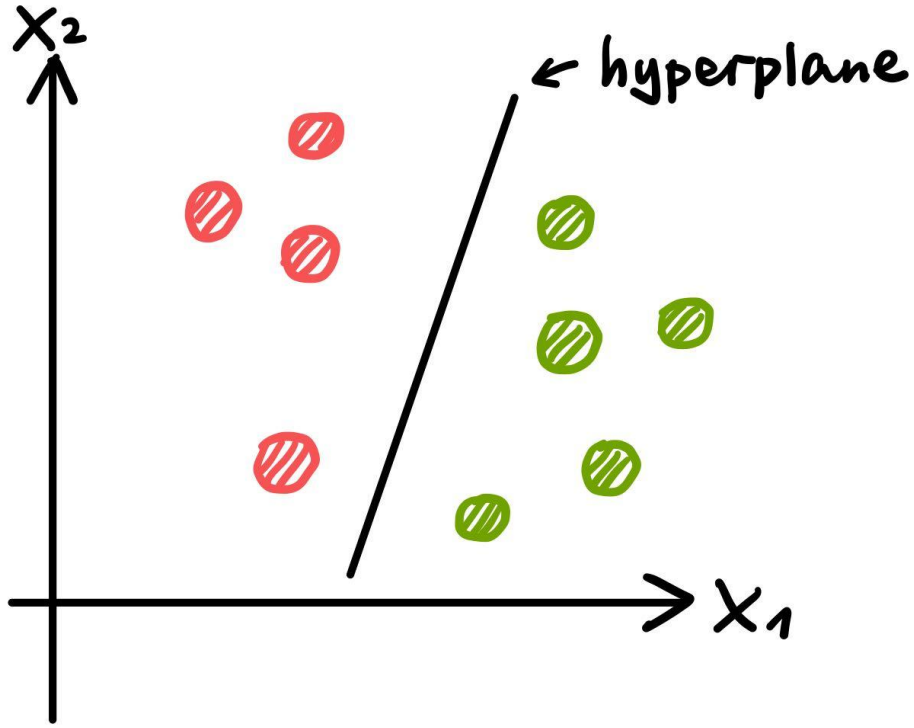
- Imagine you have a dataset with two types of data points:
- You want to classify new data points as either red or green. The main challenge is that numerous possible hyperplanes can separate the two classes, so there's a big question as we mentioned:

How to find the best hyperplane?



Main Problem: Which decision boundary ist the best one ?

How do we find it ?

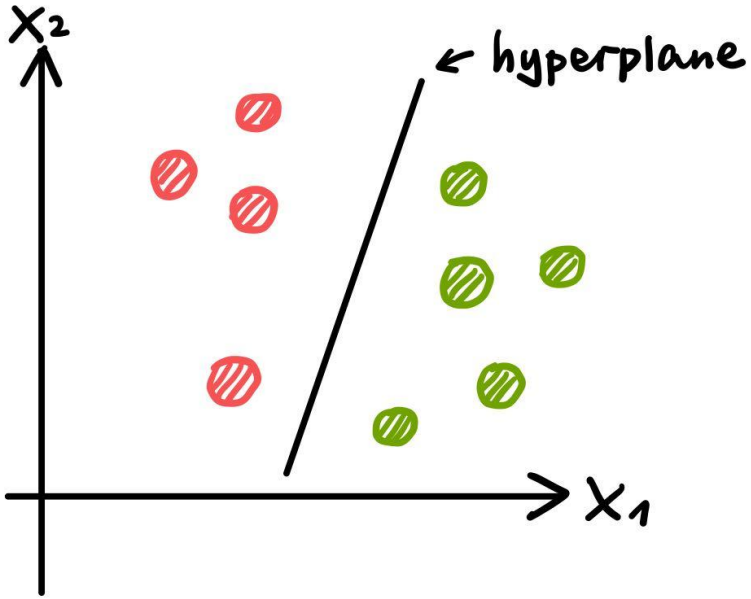


Find the one that has the maximum distance from both classes!

3. Mathematical principles of SVM

Mathematical Intuition Behind SVM

- **optimization problem**
- **given x, solve for w and b**
- **x:** data points (feature vectors)
- **w:** weight vector
- **b:** bias

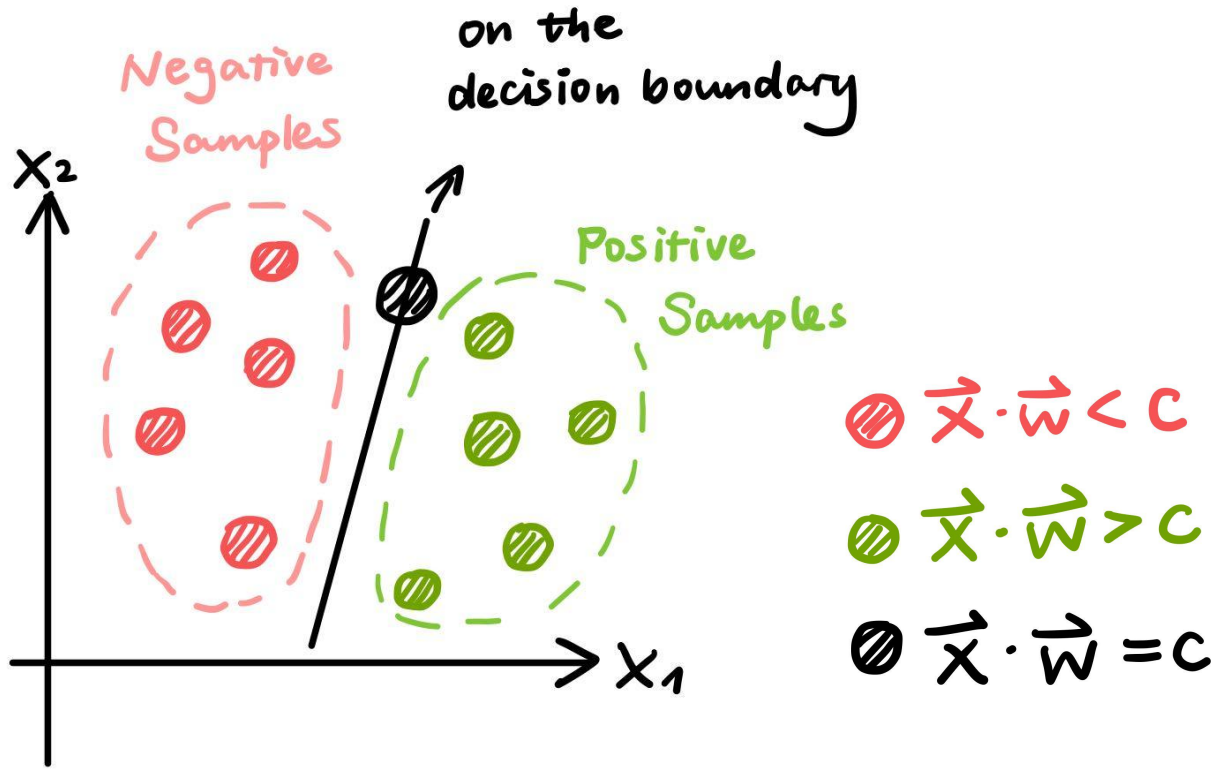


$$\vec{w} \cdot \vec{x} + b = 0$$

Step 1: Establish decision rules

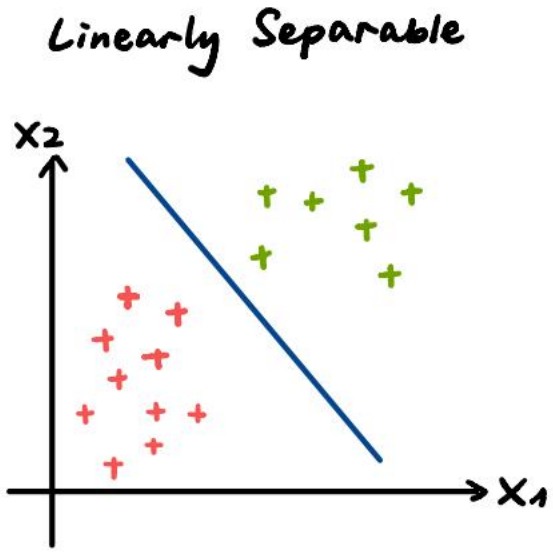
- **given a random point X:**

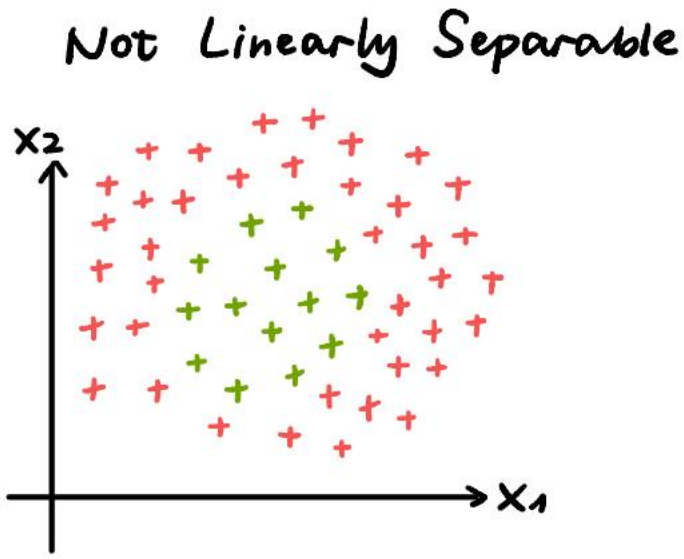
If $X \cdot W > c$ — It's a positive sample.
If $X \cdot W < c$ — It's a negative sample.
If $X \cdot W = c$ — It's on the decision boundary.



How to Solve the Problem?

- Linear separability is an idealized condition, but in reality, most data are not linearly separable.

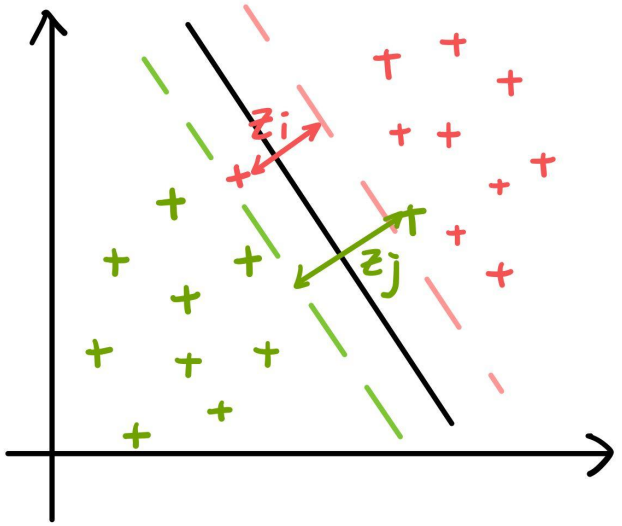


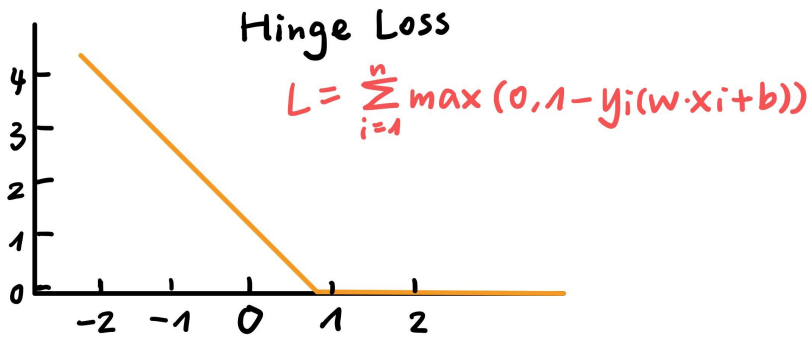


Use Soft Margin and Kernel Trick!

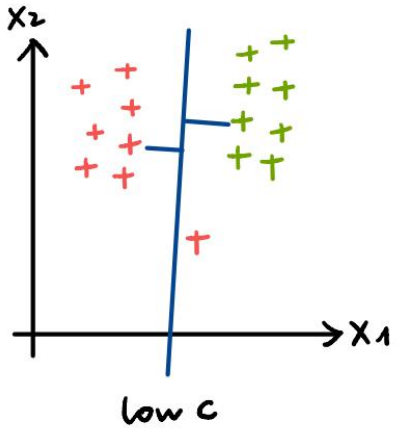
Soft Margin

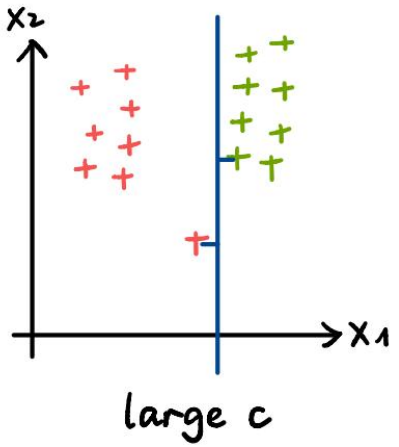
- handles **non-linearly separable** data
- slack variables z
- hinge loss
- penalty parameter c





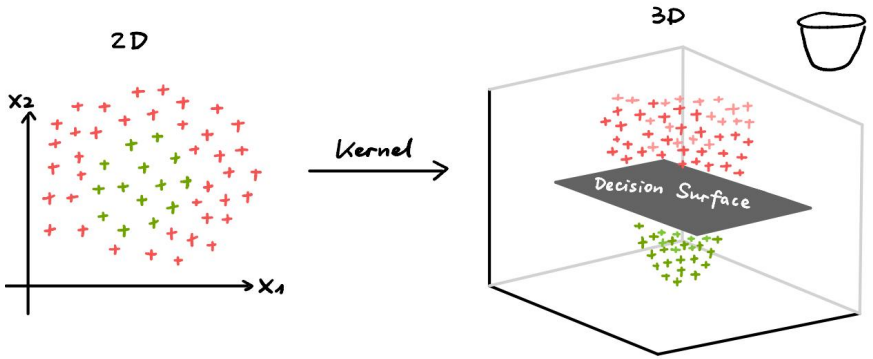
- Hyperparameter **C**



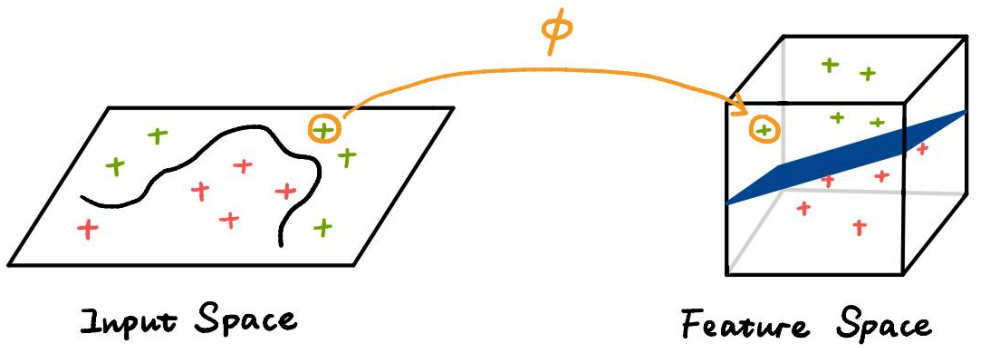


Kernel Trick

- another solution to tackle the problem of linear inseparability
- **Kernel functions** are generalized functions that take two vectors (of any dimension) as input and output a score that denotes how **similar** the input vectors are.



No need to explicitly compute the mapping to the higher-dimensional feature space.



Kernel Trick (2)

- **Common Kernel functions**

  - linear $k(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1 \cdot \mathbf{x}_2$

  - polynomial $k(\mathbf{x}_1, \mathbf{x}_2) = (\gamma \cdot (\mathbf{x}_1 \cdot \mathbf{x}_2) + c)^d$

  - Gaussian or radial basis $k(\mathbf{x}_1, \mathbf{x}_2) = \exp(-\gamma \|\mathbf{x}_1 - \mathbf{x}_2\|^2)$

  - sigmoid $k(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma \cdot (\mathbf{x}_1 \cdot \mathbf{x}_2) + c)$

- **Selecting the appropriate kernel**

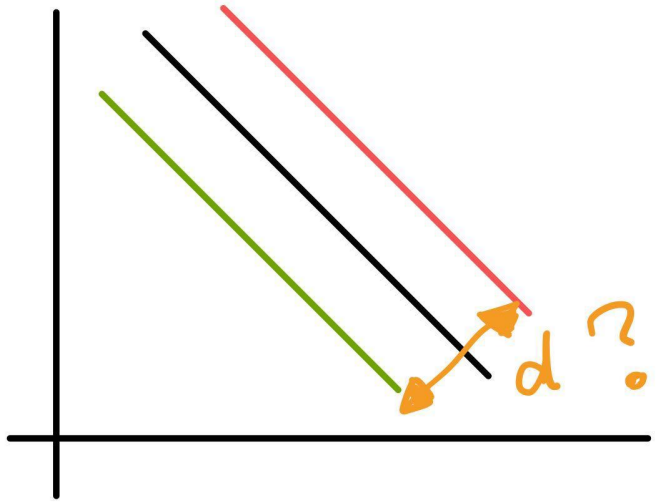
  - data complexity

  - computational resources

  - model performance

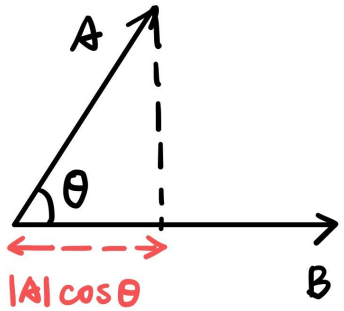
Step 2: Determine How to Find the Hyperplane

- calculate the distance (d) between the support vectors and the hyperplane (**margin**)



How? --> Use Projection!

Projection Principle

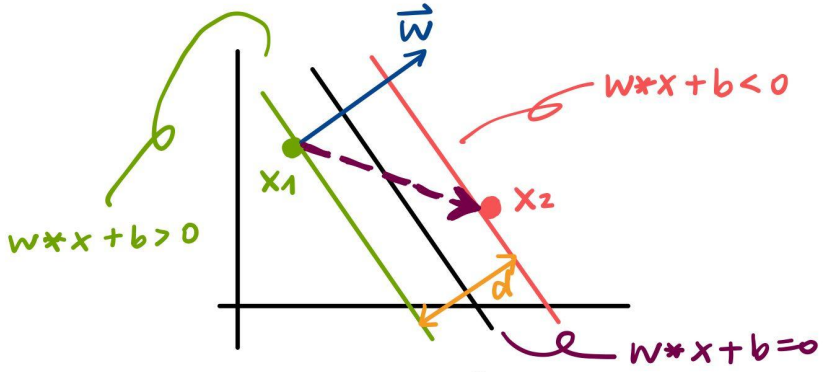


$(\vec{A} \cdot \vec{B})$ length

$$A \cdot B = |A| \cos\theta * |B|$$

$$\downarrow$$

$$|A| \cos\theta = A * \frac{B}{|B|}$$



$$d = (x_2 - x_1) \cdot \frac{\vec{w}}{\|w\|}$$

$$= \frac{x_2 \cdot \vec{w} - x_1 \cdot \vec{w}}{\|w\|}$$

Step 3: Determining Constraints

- Maximum Margin Hyperplane → $\vec{w} \cdot \vec{x} + b = 0$
- Positive Hyperplane → $\vec{w} \cdot \vec{x} + b > 0$
- Negative Hyperplane → $\vec{w} \cdot \vec{x} + b < 0$

$$\text{output } y = \begin{cases} +1 & \text{if } \vec{w} \cdot \vec{x} + b \geq 0 \\ -1 & \text{if } \vec{w} \cdot \vec{x} + b < 0 \end{cases}$$

take 2 support vectors (x_1 & x_2):

$(x_1 \rightarrow y = 1)$	$(x_2 \rightarrow y = -1)$
$1 \times (\vec{w} \cdot x_1 + b) = 1$ $+1$ $\vec{w} \cdot x_1 = 1 - b$	$-1 \times (\vec{w} \cdot x_2 + b) = 1$ -1 $\vec{w} \cdot x_2 = -b - 1$

⇓

General Constraint Equation

$$y_i (\vec{w} \cdot \vec{x} + b) \geq 1$$

Step 4: Determining the Optimization Goal

$$\text{argmax}(w^*, b^*) \frac{2}{\|w\|} \text{ such that } y_i(\vec{w} \cdot \vec{x} + b) \geq 1$$

the original optimization problem

Given:

- Hyperplane Equation: $W^T X + b = 0$
- Distance from Any Point x_i to the Hyperplane:

$$\frac{|w^T x_i + b|}{\|w\|}$$

projection value of x_i onto the hyperplane

$\|w\|$ → magnitude of $\vec{w}$

Step 4: Determining the Optimization Goal (2)

Given:

Positive Support Vector: $w^T x_i + b = 1$

Negative Support Vector: $w^T x_i + b = -1$

$\longrightarrow$ Margin $= \frac{(1)-(-1)}{\|w\|} = \frac{2}{\|w\|}$ → We have to maximize that!

$\|w\| = \sqrt{w^T w}$ → Taking the derivative of this expression is very complicated

tranform it into… $\longrightarrow$ $\frac{1}{2}\|w\|^2$

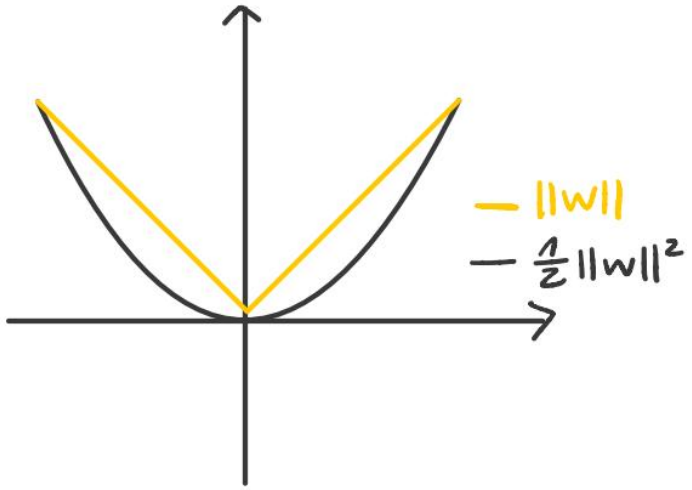
Why?

Step 4: Determining the Optimization Goal (3)

Let's take a look at the graphs of these two equations:

Both of them are **convex functions**, and it's evident that the latter is easier to differentiate and compute.

And convex functions make optimization convenient, since it has only one extremum point.



Therefore, we transform the previous equation into the **standard primal optimization problem**.

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w^T x_i + b) \geq 1$$

The primal problem

the constraints

Step 4: Determining the Optimization Goal (4)

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w^T x_i + b) \geq 1$$

The primal problem — the constraints

Or another expression:

$$f(w) = \frac{1}{2}\|w\|^2$$

subject to the constraints:

For positive class: $w^T x_i + b \geq 1$ for all i with $y_i = +1$

For negative class: $w^T x_i + b \leq -1$ for all i with $y_i = -1$

Step 4: Determining the Optimization Goal (5)

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{subject to} \quad y_i(w^T x_i + b) \geq 1$$

The primal problem — the constraints

This is a **convex optimization problem with constraints**.

Convex functions have a global optimal solution, and there is no issue of local optimal solutions.

→ Use the **Lagrange multiplier method**!

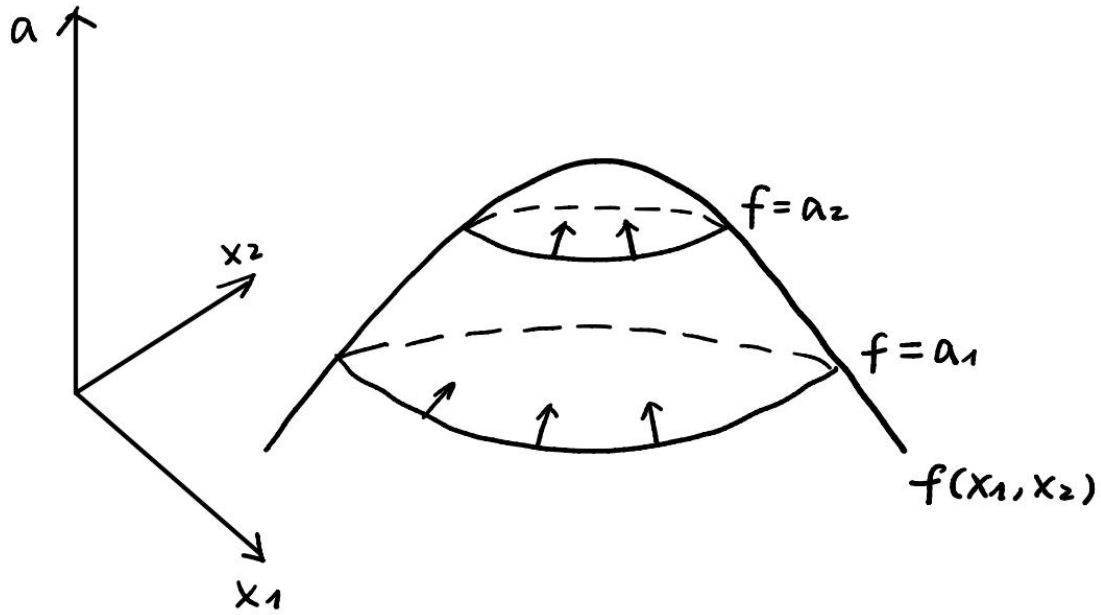
Why?

Because it was made for problems like this!

Lagrange Multiplier Method

Imagine you are standing on a level curve (f=a) of a mountain. If you want to move in the steepest direction of the slope, the curve you are on must be **perpendicular** to the path you take. This is the **gradient**:

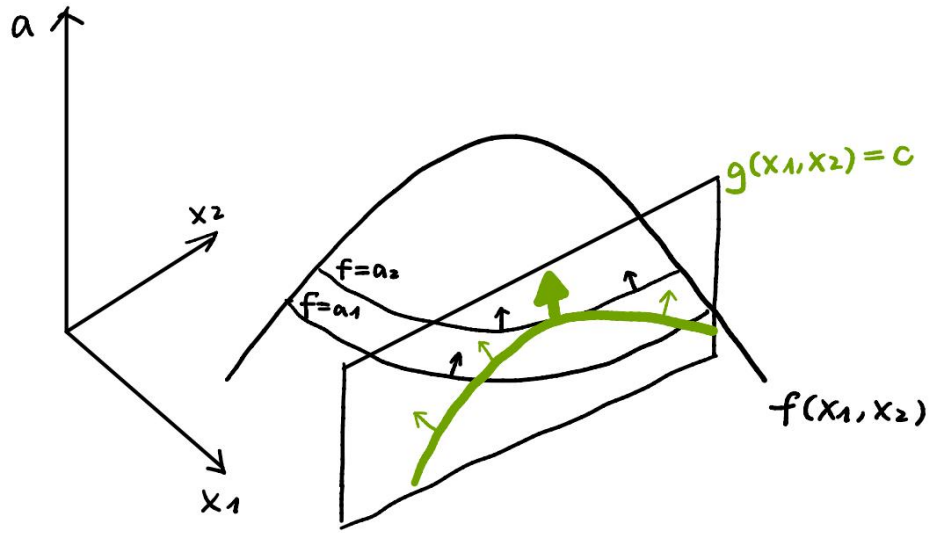
$$\nabla f$$



And: $\nabla f \perp f = a$

Lagrange Multiplier Method (2)

Now, an additional condition is introduced: there is a plane slicing through the mountain, and you can only move forward or backward along the direction defined by this plane (this is called the **constraint**). Your goal is to climb as high as possible on the mountain, but you *cannot* go higher than the point where g=c cuts into the peak.

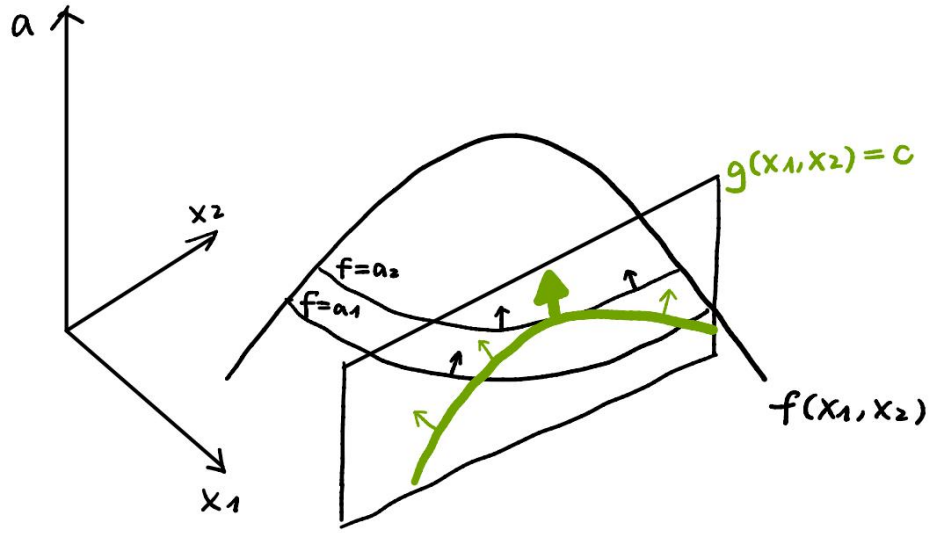


Given:

$f(x_1, x_2)$

$g = c$ (constraint)

Lagrange Multiplier Method (3)



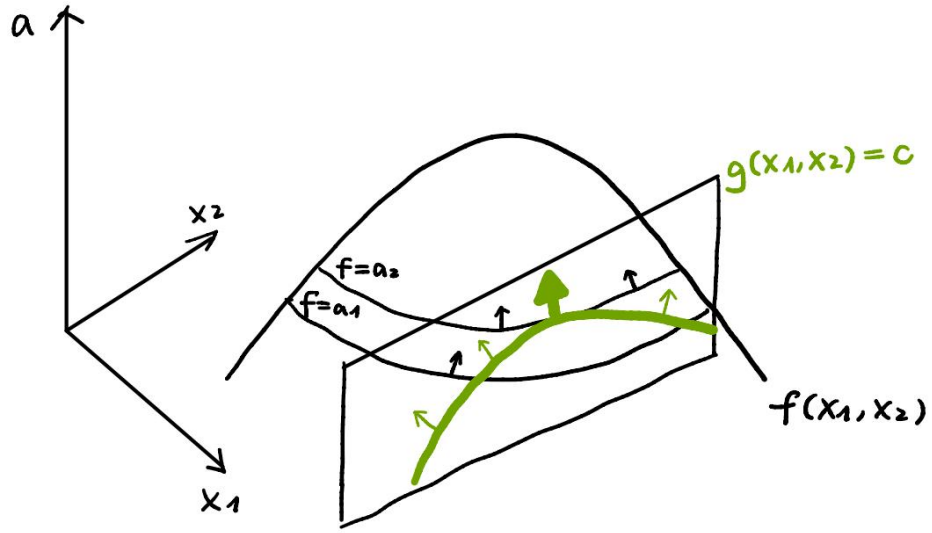
Therefore, we need to find the highest point along the boundary where the constraint intersects with the function (the green line).

When the slope of the constraint line is greater than the slope of the level curve, we move to the right;

otherwise, we move to the left.

If the two **slopes are equal**, it means we have reached the **maximum value on the constraint line**.

Lagrange Multiplier Method (4)



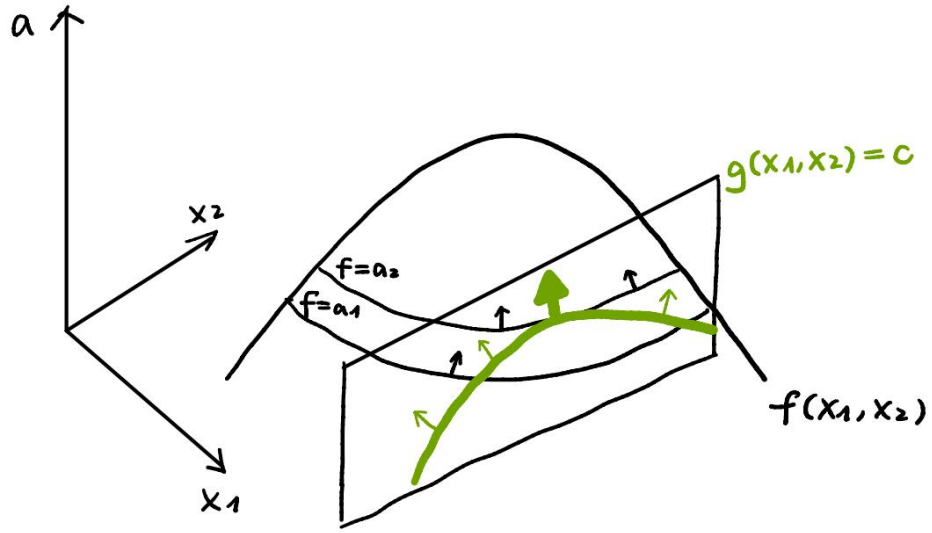
At this point, ***f = a2*** is parallel to ***g***, and both are perpendicular to the gradient of ***f***.

This means that the gradients of ***f*** and ***g*** point in the same direction, differing at most by a scalar λ.

$$f = a_2 \perp \nabla f$$

$$g \perp \nabla f$$

Lagrange Multiplier Method (5)



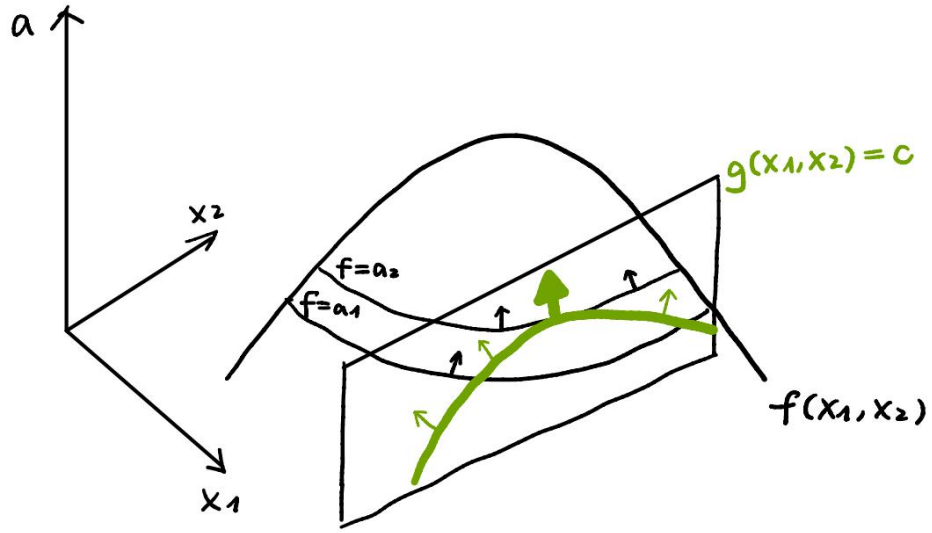
This is the conclusion we just reached:

$$\nabla f = \lambda \nabla (g)$$

Let's transform it into...

$$\nabla f - \lambda \nabla (g) = 0$$

Lagrange Multiplier Method (6)



Then, the crucial step comes:

We construct a function here!

$$L = f - \lambda(g - c)$$

Or more exactly:

$$L(x_1, x_2, \lambda) = f(x_1, x_2) - \lambda(g(x_1, x_2) - c)$$

Why?

Lagrange Multiplier Method (7)

To understand how this equation works, we can calculate the first-order derivatives of the three components of $\boldsymbol{L}$ and set them equal to zero:

$$\nabla L = \begin{bmatrix} \frac{\partial L}{\partial x_1} \\ \frac{\partial L}{\partial x_2} \\ \frac{\partial L}{\partial \lambda} \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x_1} - \lambda \frac{\partial g}{\partial x_1} \\ \frac{\partial f}{\partial x_2} - \lambda \frac{\partial g}{\partial x_2} \\ -(g(x_1, x_2) - c) \end{bmatrix} = 0$$

$$\frac{\partial f}{\partial x_1} = \nabla f$$

$$\frac{\partial g}{\partial x_1} = \nabla g$$

$$\nabla f - \lambda \nabla (g) = 0$$

You will find that the first and second elements of the vector exactly satisfy the previous conditions, as marked in red. Additionally, the third element also satisfies the constraint.

Lagrange Multiplier Method (8)

In fact, for this type of constrained optimization problem, we have a **general formula**, which is known as the **Lagrange multiplier method**. Here, $\boldsymbol{\lambda}$ is the multiplier.

$$L(x,\lambda,v)=f_0(x)+\sum_{i=1}^{M}\lambda_i f_i(x)+\sum_{i=1}^{P}v_i h_i(x)$$

Where:

$\sum_{i=1}^{P}v_i h_i(x)$: equality constraint, e.g. g=c

$\sum_{i=1}^{M}\lambda_i f_i(x)$: Inequality constraint: to standardize, we make... $\lambda_i \geq 0$. $f_i(x) \leq 0$.

The strength of the Lagrange multiplier method lies in its ability to transform a constrained optimization problem into an **unconstrained** one. We can then solve it by simply setting the gradient to zero.

Karuch-Kuhn-Tucker (KKT) Conditions

In constrained optimization problems, in addition to optimizing the objective function ***f(x)***, a set of equality and inequality constraints must also be satisfied. Directly solving such problems is very complex because:

- It requires finding the optimal solution within the “feasible region” defined by the constraints.
- The “relaxed” or “tight” state of inequality constraints can dynamically change.

The KKT conditions unify these complex concepts by providing a mathematical framework that integrates the objective function and constraint conditions, transforming them into a solvable system of equations, especially for **inequality-constrained optimization**.

Functions:

- Verification of Optimality: Determine whether a solution satisfies all requirements of the optimization problem.
- Finding the Optimal Solution: Transform the problem into a system of equations, making it easier to solve.

But what’s the “relaxed” or “tight” state of inequality constraints?

Complementary Slackness

- part of the **KKT Conditions**
- used to handle optimization problems with **inequality constraints**
- to dynamically determine which constraints are "active" (**tight** constraints) and which are "inactive" (**loose** constraints)

$$\lambda_i f_i(x) = 0 \,,\; \lambda \geq 0$$

- if $f_i(x) < 0$ (the constraint is strictly satisfied)

  → $\lambda_i = 0$ (the constraint does not affect the optimization)

- if $f_i(x) = 0$ (the constraint just reaches the boundary)

  → $\lambda_i > 0$ (the constraint actively affect the optimization)

- if $f_i(x) > 0$ (violates the constraint)

  → not allowed, as it contradicts the constraints.

Primal Feasibility

- part of the **KKT Conditions**
- determine whether the solution to the primal problem satisfies all the constraints

$$f_i(x) \leq 0, \quad h_j(x) = 0.$$

Dual Feasibility

- part of the **KKT Conditions**
- related to the dual problem

$$\lambda_i \geq 0$$

Stationarity

- part of the **KKT Conditions**
- ensures that the gradients of the objective function and the constraints are linearly related
- provides a necessary condition for the optimal solution of the optimization problem

$$\nabla f(x) + \sum_i \lambda_i \nabla h_i(x) + \sum_j v_j \nabla g_j(x) = 0$$

Karuch-Kuhn-Tucker (KKT) Conditions (2)

- Primal Feasibility

$$f_i(x) \leq 0, \quad h_j(x) = 0.$$

- Dual Feasibility

$$\lambda_i \geq 0$$

- Complementary Slackness

$$\lambda_i f_i(x) = 0 \quad \forall i$$

- Stationarity

$$\nabla f(x) + \sum_i \lambda_i \nabla h_i(x) + \sum_j v_j \nabla g_j(x) = 0$$

Based on the KKT conditions, we can solve for the final decision hyperplane.

However, in the SVM model, for efficiency and to facilitate the use of the kernel trick, the primal problem is often transformed into its corresponding **dual problem** for solving.

4. Implementation of SVM in Python

- Letter
  - Author identification
  - Language identification
- News
- Sentiment

Letter

{'author': 'Virginia Woolf', 'year': '1908', 'lang': 'en', 'text': 'I dreamt of Clarissa last night; she was new born, and had a fine row of teeth, which were without roots; and she could say ‘no objection’ which I thought proved something out of Moore. Yr. B.', 'file': 'woolf/json/letter_450.json'}

```
from collections import Counter
import json
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

train_file_path = …
evaluate_file_path = …

def load_data(file_path):
    …
    return texts, authors, langs


train_texts, train_authors, train_langs = load_data(train_file_path)

eval_texts, eval_authors, eval_langs = load_data(evaluate_file_path)

vectorizer = TfidfVectorizer(max_features=5000)
X_train = vectorizer.fit_transform(train_texts)
X_eval = vectorizer.transform(eval_texts)
```

{'author': 'Virginia Woolf', 'year': '1908', 'lang': 'en', 'text': 'I dreamt of Clarissa last night; she was new born, and had a fine row of teeth, which were without roots; and she could say ‘no objection’ which I thought proved something out of Moore. Yr. B.', 'file': 'woolf/json/letter_450.json'}

```
#Author Classification

svc_author = SVC(kernel='linear', C=1.0)

svc_author.fit(X_train, train_authors)

y_pred_eval_author = svc_author.predict(X_eval)

print("Author Report:")
print(classification_report(eval_authors, y_pred_eval_author))
```

```
#Language Classification

svc_lang = SVC(kernel='linear', C=1.0, class_weight='balanced')

svc_lang.fit(X_train, train_langs)

y_pred_eval_lang = svc_lang.predict(X_eval)

target_classes = list(Counter(eval_langs).keys())

print("Language Report:")
print(classification_report(eval_langs, y_pred_eval_lang,
labels=target_classes))
```

Author Report:

	precision	recall	f1-score	support
Franz Kafka	0.88	0.90	0.89	280
Friedrich Schiller	0.72	0.79	0.76	266
Henrik Ibsen	1.00	0.98	0.99	897
James Joyce	0.94	0.92	0.93	682
Johann Wolfgang von Goethe	0.73	0.64	0.68	228
Virginia Woolf	0.97	0.98	0.98	1901
Wilhelm Busch	0.93	0.95	0.94	627
accuracy			0.94	4881
macro avg	0.88	0.88	0.88	4881
weighted avg	0.94	0.94	0.94	4881

- Weighted Precision = $\frac{\sum_{i=1}^{k}(\text{Precision}_i \times \text{Support}_i)}{\sum_{i=1}^{k}\text{Support}_i}$
- Weighted Recall = $\frac{\sum_{i=1}^{k}(\text{Recall}_i \times \text{Support}_i)}{\sum_{i=1}^{k}\text{Support}_i}$
- Weighted F1-Score = $\frac{\sum_{i=1}^{k}(\text{F1}_i \times \text{Support}_i)}{\sum_{i=1}^{k}\text{Support}_i}$

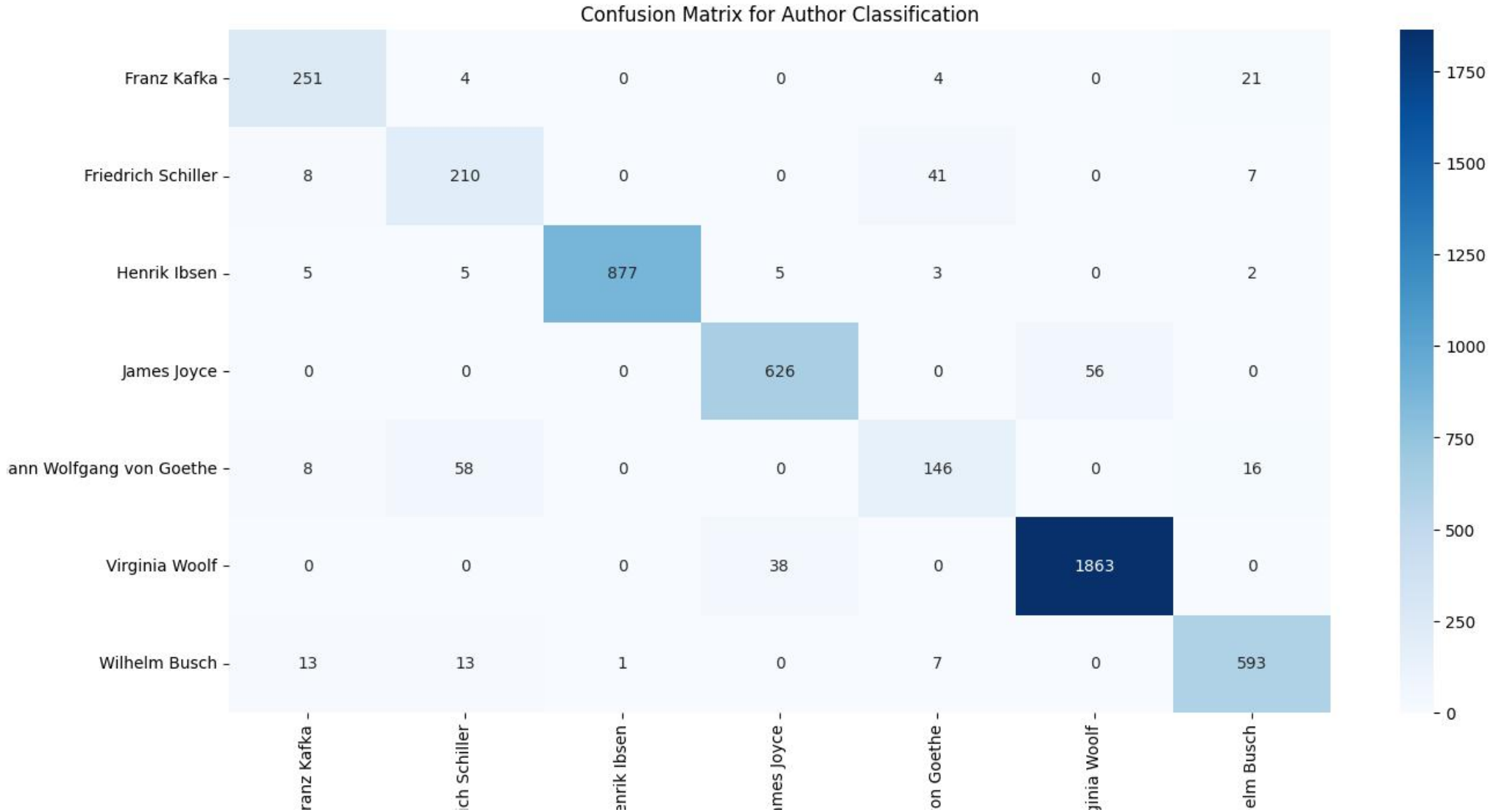
Language Report:

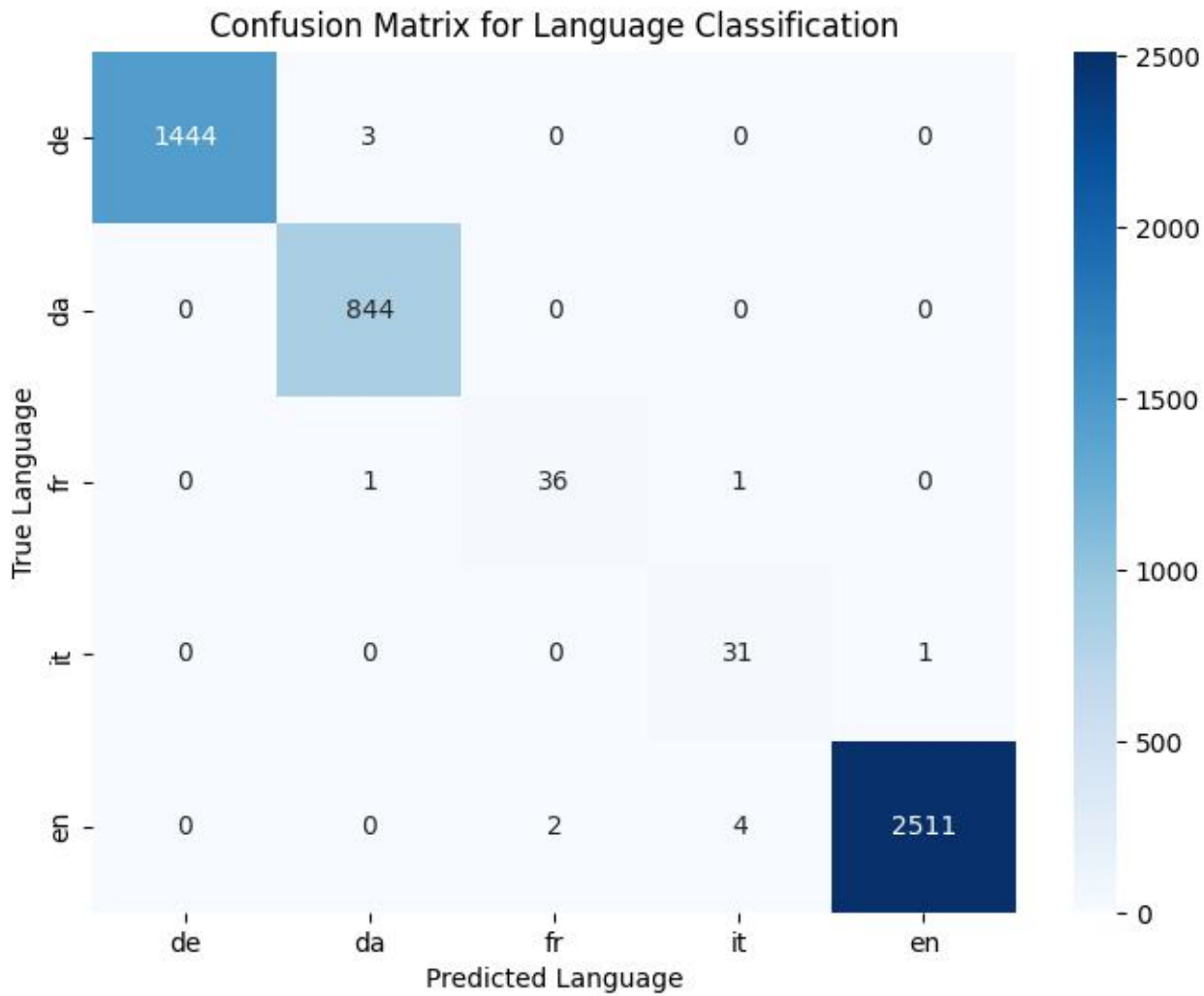
	precision	recall	f1-score	support
de	1.00	1.00	1.00	1448
da	1.00	1.00	1.00	844
fr	0.95	0.95	0.95	38
it	0.86	0.97	0.91	32
en	1.00	1.00	1.00	2519
micro avg	1.00	1.00	1.00	4881
macro avg	0.96	0.98	0.97	4881
weighted avg	1.00	1.00	1.00	4881

$$\text{Precision (Micro)} = \frac{\text{Total True Positives (TP)}}{\text{Total True Positives (TP)} + \text{Total False Positives (FP)}}$$

$$\text{Recall (Micro)} = \frac{\text{Total True Positives (TP)}}{\text{Total True Positives (TP)} + \text{Total False Negatives (FN)}}$$

$$\text{F1-score (Micro)} = 2 \times \frac{\text{Precision (Micro)} \times \text{Recall (Micro)}}{\text{Precision (Micro)} + \text{Recall (Micro)}}$$


Confusion Matrix for Author Classification
Franz Kafka
Friedrich Schiller
Henrik Ibsen
James Joyce
ann Wolfgang von Goethe
Virginia Woolf
Wilhelm Busch
251 4 0 0 4 0 21
8 210 0 0 41 0 7
5 5 877 5 3 0 2
0 0 0 626 0 56 0
8 58 0 0 146 0 16
0 0 0 38 0 1863 0
13 13 1 0 7 0 593
ranz Kafka
ich Schiller
enrik Ibsen
mes Joyce
on Goethe
ginia Woolf
elm Busch
1750
1500
1250
1000
750
500
250
0


Confusion Matrix for Language Classification
True Language
Predicted Language
de
da
fr
it
en
1444
3
0
0
0
0
844
0
0
0
0
1
36
1
0
0
0
0
31
1
0
0
2
4
2511
2500
2000
1500
1000
500
0

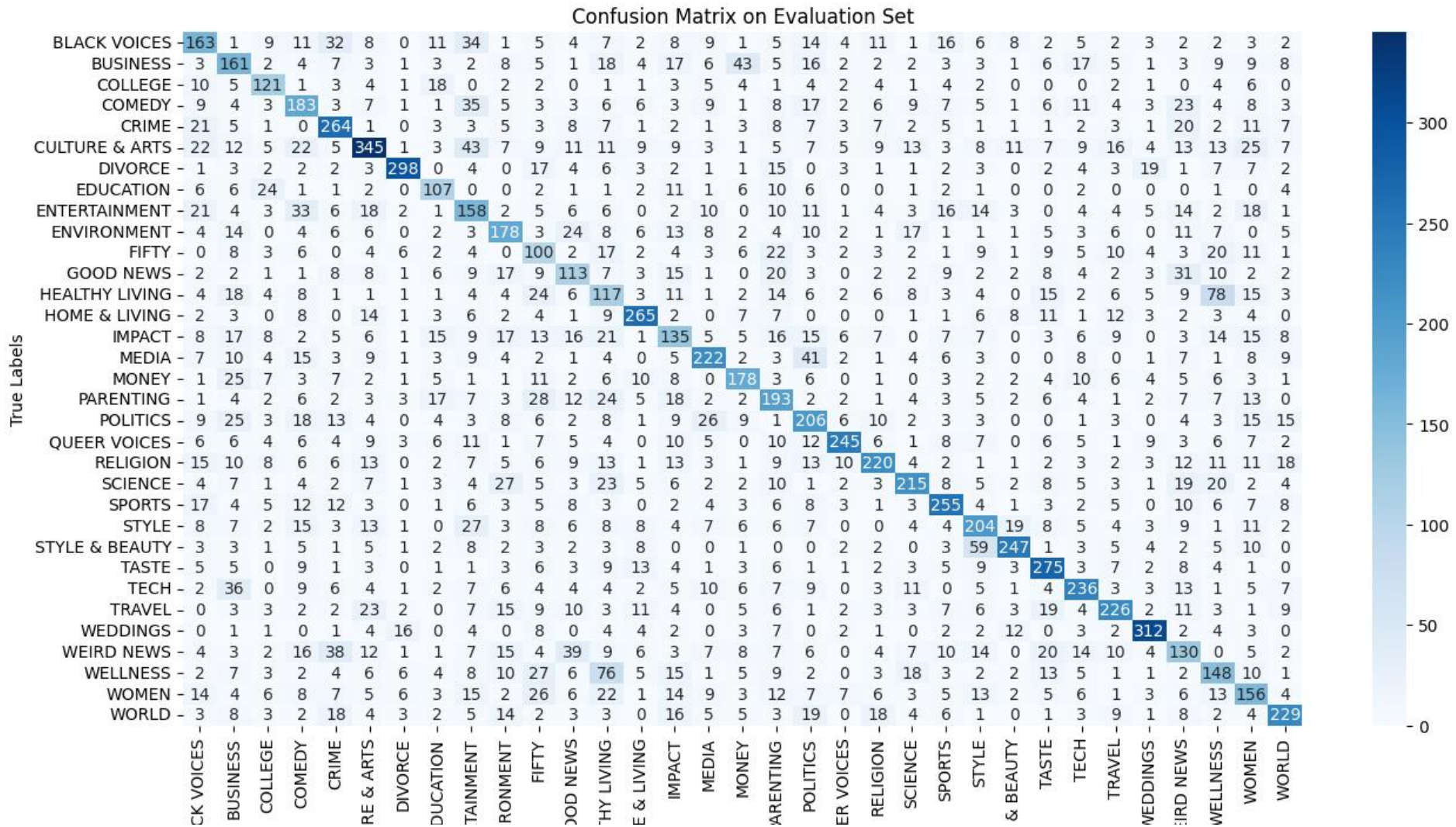
News

{'category': 'CULTURE & ARTS', 'headline': 'Illustrator Draws 100 Happy Things To Get Over A Breakup', 'authors': 'Maddie Crum', 'link': 'https://www.huffingtonpost.com/entry/rinee-shah-100-happy-things_us_560c37fee4b0dd85030a63f3', 'short_description': 'A little art therapy can go a long way.', 'date': '2015-10-02'}

{'category': 'FIFTY', 'headline': '6 Foods That Fight Pain Naturally', 'authors': 'Yagana Shah', 'link': 'https://www.huffingtonpost.com/entry/foods-for-pain_n_5411711.html', 'short_description': '', 'date': '2014-06-01'}

Report of News Classification:

	precision	recall	f1-score	support
BLACK VOICES	0.43	0.42	0.42	392
BUSINESS	0.37	0.42	0.40	380
COLLEGE	0.50	0.57	0.53	212
COMEDY	0.43	0.46	0.44	399
CRIME	0.56	0.65	0.60	409
CULTURE & ARTS	0.62	0.51	0.56	673
DIVORCE	0.83	0.71	0.77	419
EDUCATION	0.46	0.54	0.50	198
ENTERTAINMENT	0.35	0.41	0.38	387
ENVIRONMENT	0.48	0.50	0.49	355
FIFTY	0.27	0.37	0.31	273
GOOD NEWS	0.35	0.37	0.36	305
HEALTHY LIVING	0.25	0.30	0.27	386
HOME & LIVING	0.68	0.69	0.68	386
IMPACT	0.36	0.34	0.35	400
MEDIA	0.60	0.56	0.58	395
MONEY	0.55	0.55	0.55	324
PARENTING	0.43	0.49	0.46	391
POLITICS	0.45	0.49	0.47	420
QUEER VOICES	0.77	0.59	0.67	415
RELIGION	0.63	0.50	0.56	440
SCIENCE	0.62	0.52	0.56	414
SPORTS	0.62	0.62	0.62	410
STYLE	0.49	0.49	0.49	413
STYLE & BEAUTY	0.74	0.63	0.68	391
TASTE	0.61	0.69	0.65	397
TECH	0.60	0.57	0.58	416
TRAVEL	0.61	0.56	0.58	405
WEDDINGS	0.77	0.78	0.77	400
WEIRD NEWS	0.33	0.32	0.32	408
WELLNESS	0.35	0.36	0.36	407
WOMEN	0.38	0.39	0.39	400
WORLD	0.63	0.57	0.60	404
accuracy			0.52	12824
macro avg	0.52	0.51	0.51	12824
weighted avg	0.53	0.52	0.52	12824

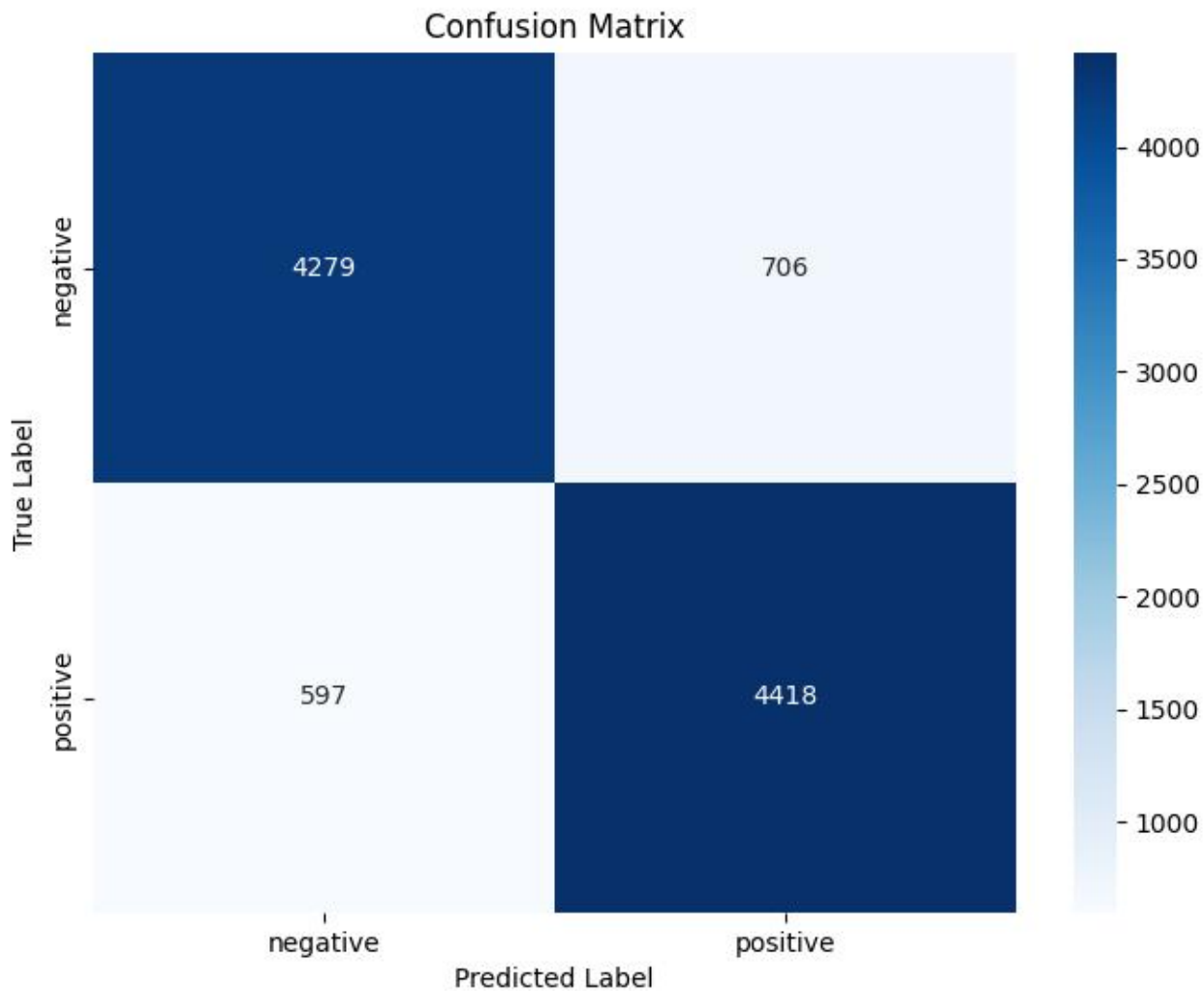

Confusion Matrix on Evaluation Set
True Labels
BLACK VOICES 163 1 9 11 32 8 0 11 34 1 5 4 7 2 8 9 1 5 14 4 11 1 16 6 8 2 5 2 3 2 2 3 2
BUSINESS 3 161 2 4 7 3 1 3 2 8 5 1 18 4 17 6 43 5 16 2 2 2 3 3 1 6 17 5 1 3 9 9 8
COLLEGE 10 5 121 1 3 4 1 18 0 2 2 0 1 1 3 5 4 1 4 2 4 1 4 2 0 0 0 2 1 0 4 6 0
COMEDY 9 4 3 183 3 7 1 1 35 5 3 3 6 6 3 9 1 8 17 2 6 9 7 5 1 6 11 4 3 23 4 8 3
CRIME 21 5 1 0 264 1 0 3 3 5 3 8 7 1 2 1 3 8 7 3 7 2 5 1 1 1 2 3 1 20 2 11 7
CULTURE & ARTS 22 12 5 22 5 345 1 3 43 7 9 11 11 9 9 3 1 5 7 5 9 13 3 8 11 7 9 16 4 13 13 25 7
DIVORCE 1 3 2 2 2 3 298 0 4 0 17 4 6 3 2 1 1 15 0 3 1 1 2 3 0 2 4 3 19 1 7 7 2
EDUCATION 6 6 24 1 1 2 0 107 0 0 2 1 1 2 11 1 6 10 6 0 0 1 2 1 0 0 2 0 0 0 1 0 4
ENTERTAINMENT 21 4 3 33 6 18 2 1 158 2 5 6 6 0 2 10 0 10 11 1 4 3 16 14 3 0 4 4 5 14 2 18 1
ENVIRONMENT 4 14 0 4 6 6 0 2 3 178 3 24 8 6 13 8 2 4 10 2 1 17 1 1 1 5 3 6 0 11 7 0 5
FIFTY 0 8 3 6 0 4 6 2 4 0 100 2 17 2 4 3 6 22 3 2 3 2 1 9 1 9 5 10 4 3 20 11 1
GOOD NEWS 2 2 1 1 8 8 1 6 9 17 9 113 7 3 15 1 0 20 3 0 2 2 9 2 2 8 4 2 3 31 10 2 2
HEALTHY LIVING 4 18 4 8 1 1 1 1 4 4 24 6 117 3 11 1 2 14 6 2 6 8 3 4 0 15 2 6 5 9 78 15 3
HOME & LIVING 2 3 0 8 0 14 1 3 6 2 4 1 9 265 2 0 7 7 0 0 0 1 1 6 8 11 1 12 3 2 3 4 0
IMPACT 8 17 8 2 5 6 1 15 9 17 13 16 21 1 135 5 5 16 15 6 7 0 7 7 0 3 6 9 0 3 14 15 8
MEDIA 7 10 4 15 3 9 1 3 9 4 2 1 4 0 5 222 2 3 41 2 1 4 6 3 0 0 8 0 1 7 1 8 9
MONEY 1 25 7 3 7 2 1 5 1 1 11 2 6 10 8 0 178 3 6 0 1 0 3 2 2 4 10 6 4 5 6 3 1
PARENTING 1 4 2 6 2 3 3 17 7 3 28 12 24 5 18 2 2 193 2 2 1 4 3 5 2 6 4 1 2 7 7 13 0
POLITICS 9 25 3 18 13 4 0 4 3 8 6 2 8 1 9 26 9 1 206 6 10 2 3 3 0 0 1 3 0 4 3 15 15
QUEER VOICES 6 6 4 6 4 9 3 6 11 1 7 5 4 0 10 5 0 10 12 245 6 1 8 7 0 6 5 1 9 3 6 7 2
RELIGION 15 10 8 6 6 13 0 2 7 5 6 9 13 1 13 3 1 9 13 10 220 4 2 1 1 2 3 2 3 12 11 11 18
SCIENCE 4 7 1 4 2 7 1 3 4 27 5 3 23 5 6 2 2 10 1 2 3 215 8 5 2 8 5 3 1 19 20 2 4
SPORTS 17 4 5 12 12 3 0 1 6 3 5 8 3 0 2 4 3 6 8 3 1 3 255 4 1 3 2 5 0 10 6 7 8
STYLE 8 7 2 15 3 13 1 0 27 3 8 6 8 8 4 7 6 6 7 0 0 4 4 204 19 8 5 4 3 9 1 11 2
STYLE & BEAUTY 3 3 1 5 1 5 1 2 8 2 3 2 3 8 0 0 1 0 0 2 2 0 3 59 247 1 3 5 4 2 5 10 0
TASTE 5 5 0 9 1 3 0 1 1 3 6 3 9 13 4 1 3 6 1 1 2 3 5 9 3 275 3 7 2 8 4 1 0
TECH 2 36 0 9 6 4 1 2 7 6 4 4 4 2 5 10 6 7 9 0 3 11 0 5 1 4 236 3 3 13 1 5 7
TRAVEL 0 3 3 2 2 23 2 0 7 15 9 10 3 11 4 0 5 6 1 2 3 3 7 6 3 19 4 226 2 11 3 1 9
WEDDINGS 0 1 1 0 1 4 16 0 4 0 8 0 4 4 2 0 3 7 0 2 1 0 2 2 12 0 3 2 312 2 4 3 0
WEIRD NEWS 4 3 2 16 38 12 1 1 7 15 4 39 9 6 3 7 8 7 6 0 4 7 10 14 0 20 14 10 4 130 0 5 2
WELLNESS 2 7 3 2 4 6 6 4 8 10 27 6 76 5 15 1 5 9 2 0 3 18 3 2 2 13 5 1 1 2 148 10 1
WOMEN 14 4 6 8 7 5 6 3 15 2 26 6 22 1 14 9 3 12 7 7 6 3 5 13 2 5 6 1 3 6 13 156 4
WORLD 3 8 3 2 18 4 3 2 5 14 2 3 3 0 16 5 5 3 19 0 18 4 6 1 0 1 3 9 1 8 2 4 229
CK VOICES
BUSINESS
COLLEGE
COMEDY
CRIME
RE & ARTS
DIVORCE
DUCATION
TAINMENT
RONMENT
FIFTY
OOD NEWS
THY LIVING
E & LIVING
IMPACT
MEDIA
MONEY
ARENTING
POLITICS
ER VOICES
RELIGION
SCIENCE
SPORTS
STYLE
& BEAUTY
TASTE
TECH
TRAVEL
WEDDINGS
EIRD NEWS
WELLNESS
WOMEN
WORLD
300
250
200
150
100
50
0

Sentiment

{'text': "This film is the worst film I have ever seen. The story line is weak - I couldn't even follow it. The acting is high-schoolish. The sound track is irritating. The attempts at humor are not. The editing is horrible. The credits are even slow - I would be embarrassed to have my name associated with this waste of film. Don't waste your time even thinking about this attempt at acting.", 'sentiment': 'negative'}

Sentiment Report:

	precision	recall	f1-score	support
negative	0.88	0.86	0.87	4985
positive	0.86	0.88	0.87	5015
accuracy			0.87	10000
macro avg	0.87	0.87	0.87	10000
weighted avg	0.87	0.87	0.87	10000


Confusion Matrix
4279
706
597
4418
negative
positive
True Label
negative
positive
Predicted Label
4000
3500
3000
2500
2000
1500
1000

5. Advantages and Disadvantages

Advantages

1. Effective in High-Dimensional Spaces:
2. Robust to Overfitting:
3. Works Well with Small Sample Sizes

Disadvantages

1. Computationally Intensive:
2. Not Suitable for Large Datasets:
3. Sensitive to Parameter Selection:

6. Development Trends of SVM

1. Integration with Deep Learning

2. Innovations in Kernel Functions
   - Adaptive Kernel
   - Graph-Based Kernel
   - Multiple Kernel Learning

3. Expansion of Application Domains
   - Bioinformatics
   - Financial Analysis
   - Medical Diagnostics
   - Network Security

https://www.pinterest.com/pin/machine-learningneural-networks--95912667053739435/

References

Dhiraj K. Top 4 advantages and disadvantages of Support Vector Machine or SVM. In: https://dhirajkumarblog.medium.com/top-4-advantages-and-disadvantages-of-support-vector-machine-or-svm-a3c06a2b107, 06.12.2024

Jangdaehan. The Future of Support Vector Machines: Trends and Innovations in Supervised Learning. In: https://medium.com/@jangdaehan1/the-future-of-support-vector-machines-trends-and-innovations-in-supervised-learning-fcb95bb77b53, 06.12.2024

https://readmedium.com/support-vector-machines-svm-ml-basics-machine-learning-data-science-getting-started-1683fc99cd45

https://medium.com/@andrew.chamberlain/a-simple-explanation-of-why-lagrange-multipliers-works-253e2cdcbf74