Machine Learning

MUSIC GENRE CLASSIFICATION

Quach Tuan Anh - 20225469
Pham Tran Tuan Khang - 20225503
Dinh Van Kien - 20225505
Nguyen Tran Nghia - 20225452
Vu Ngoc Ha - 20225490

OVERVIEW

INTRODUCTION

INTRODUCTION

Problem Statement	The challenge of organizing and categorizing vast digital music libraries.
Objective	To develop a robust and accurate music genre classification system using machine learning.
Methodology	Data collection, feature extraction, model training, and application development.
Results	Utilization of advanced techniques such as ensemble learning for improved accuracy and usability through local and web-based applications.

DATA PREPARATION

DATA PREPARATION

ANDRADA · UPDATED 4 YEARS AGO

748 | New Notebook | Download (1 GB)

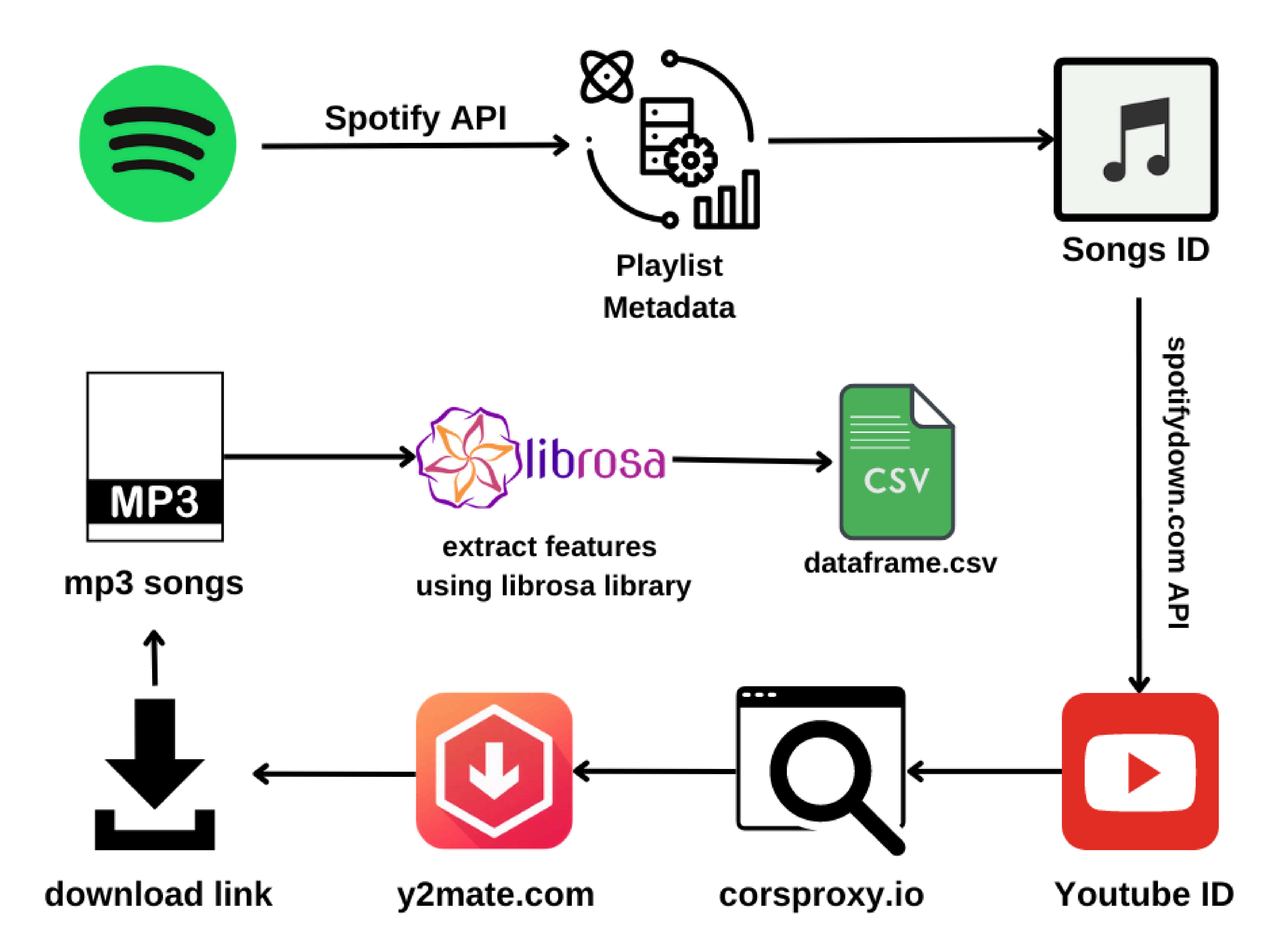
GTZAN Dataset - Music Genre Classification

Audio Files | Mel Spectrograms | CSV with extracted features

Data Card | Code (225) | Discussion (17) | Suggestions (0)

- A collection of 10 genres with 100 audio files each, all having a length of 30 seconds
- Consists of 57 attributes

DATA PREPARATION



DATA PREPROCESSING

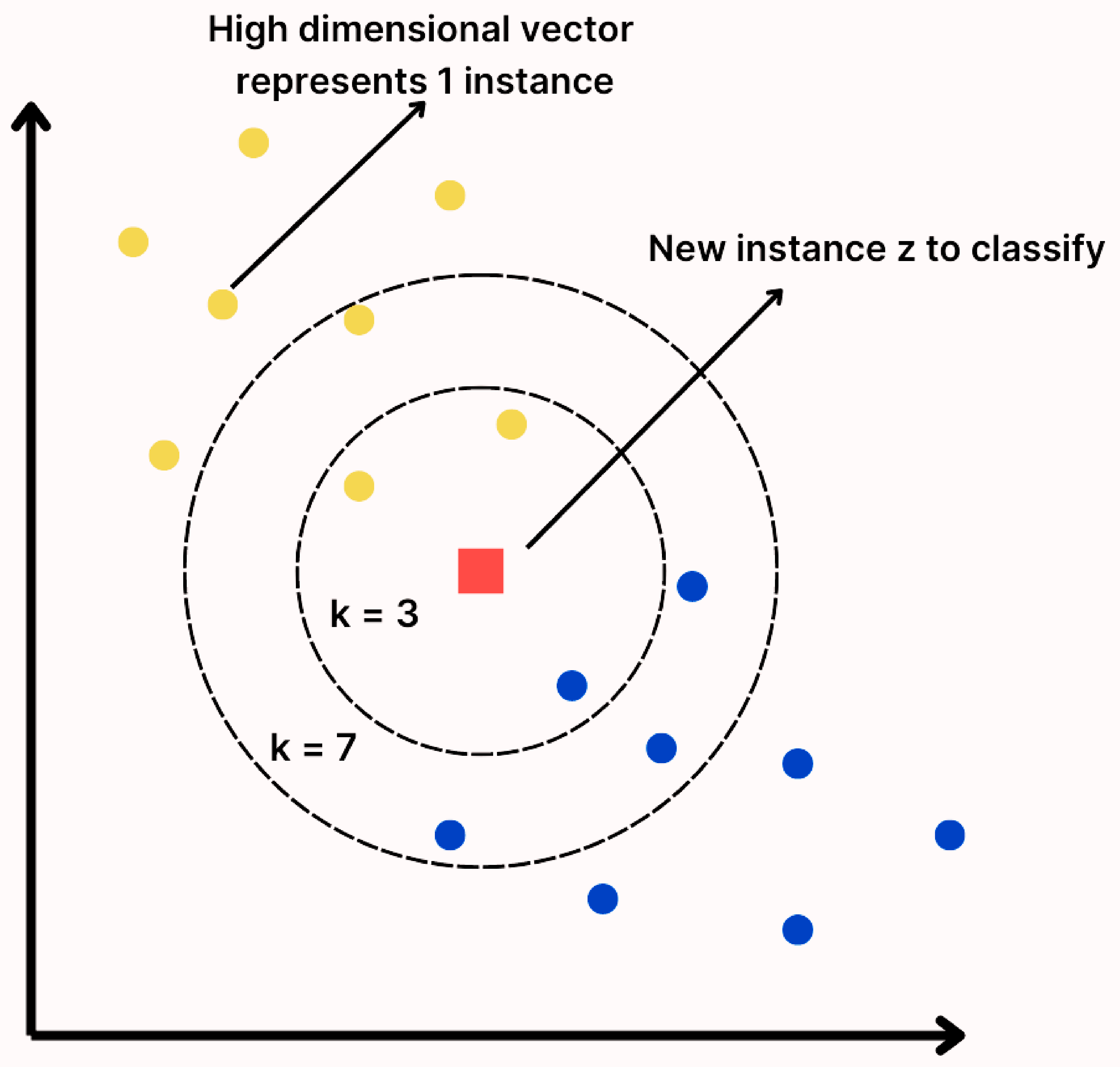
- Importing libraries and loading dataset.
- Separating features and target variables.
- Label encoding.
- Normalizing features.
- Splitting the Data into Training and Test Sets.

MODELLING


44 function
var width
45 if(width < 750){
cardssmallscreen();
46 }else{
47 cardsbigscreen();

1 K-NEAREST NEIGHBOR

Our initial approach for solving the problem, because of its simplicity and adaptability.



1

K-NEAREST NEIGHBOR

For better classification performance **=>** Adding **distance weights**.

=> Data points that are farther from the point being considered will have smaller **weights**. The weight formula W based on the distance d between 2 instances which are in consideration can be expressed as follows:

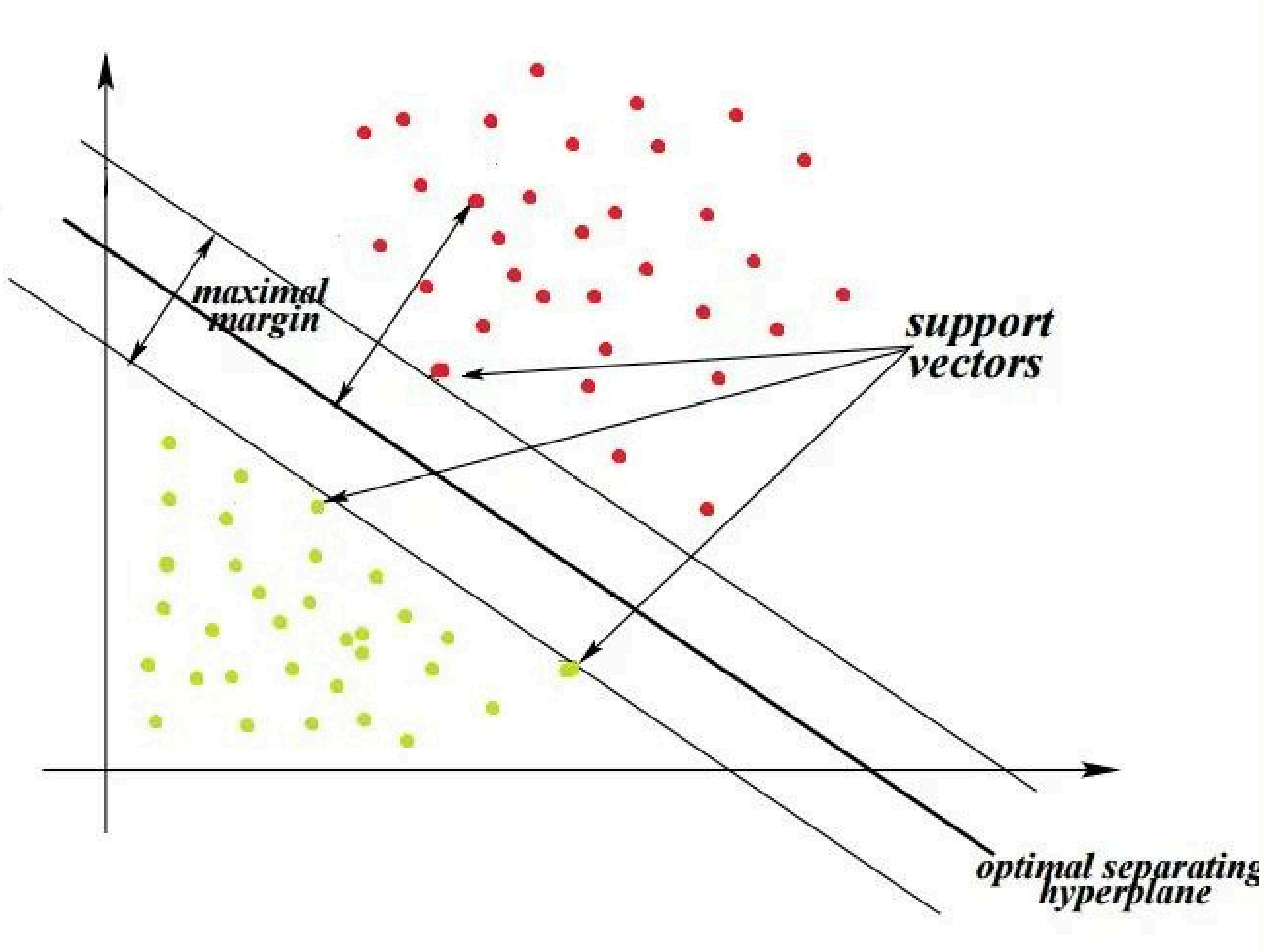
$$W = \frac{1}{d^2}$$

=> Closer neighbors should have more effects while farther instances should have less effects on the classification of new instance **z**.

SUPPORT VECTOR MACHINE

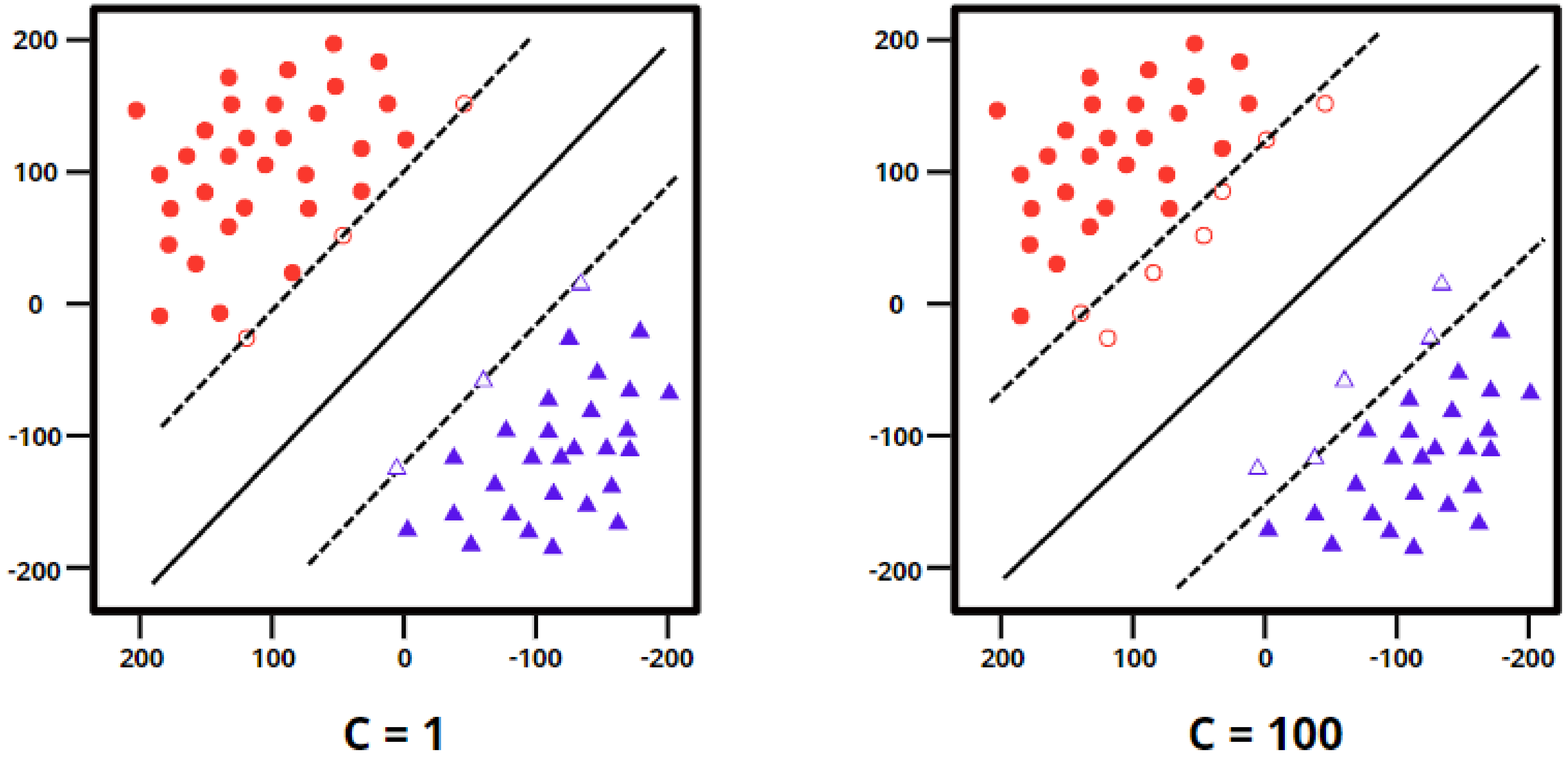
- Second approach for solving the problem due to SVM's capability of dealing with dealing with datasets with a high number of dimensions

- For data that is not linearly separable, SVM employs kernel functions to project the data into a higher-dimensional space where a hyperplane can effectively separate the classes. Common kernel functions include: Linear Kernel, Polynomial Kernel, Radial Basis Function (RBF) Kernel, Sigmoid Kernel



2

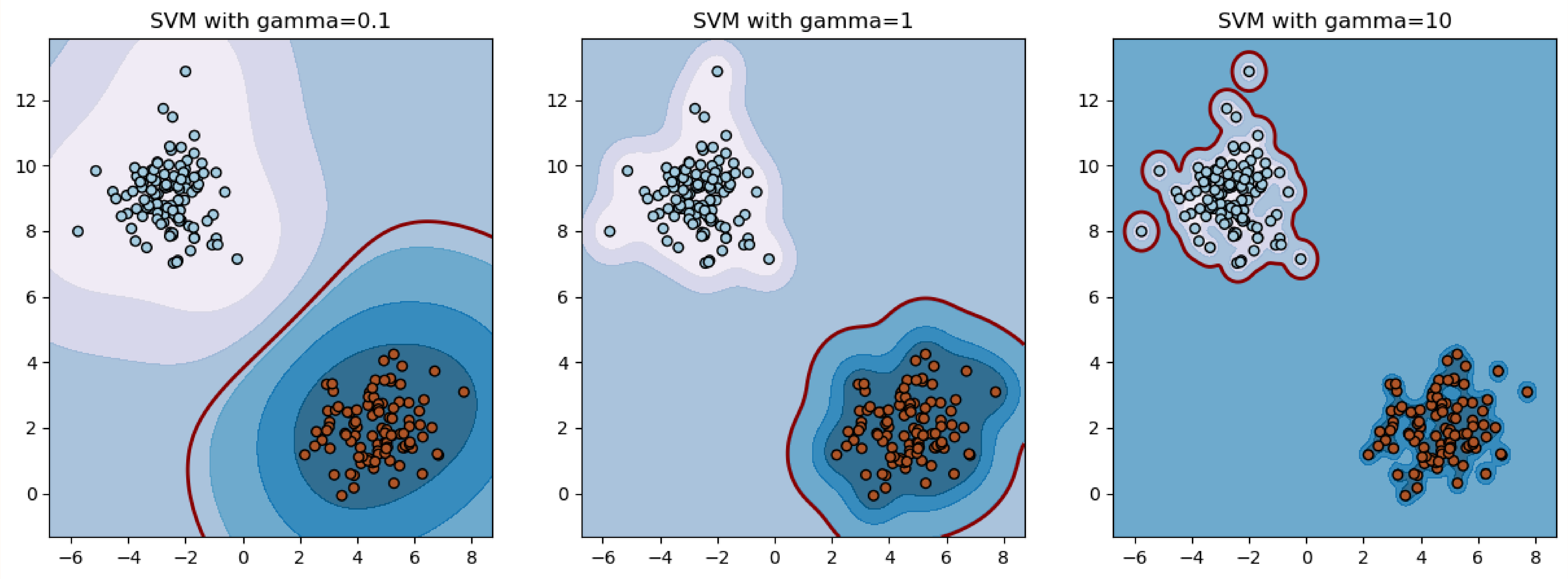
SUPPORT VECTOR MACHINE



Key hyperparameters include the regularization parameter C and the parameters specific to the chosen kernel. The regularization parameter C plays a crucial role in balancing the desire to minimize training error with the need to avoid overfitting on the test data, ultimately impacting the margin of the classifier.

SUPPORT VECTOR MACHINE

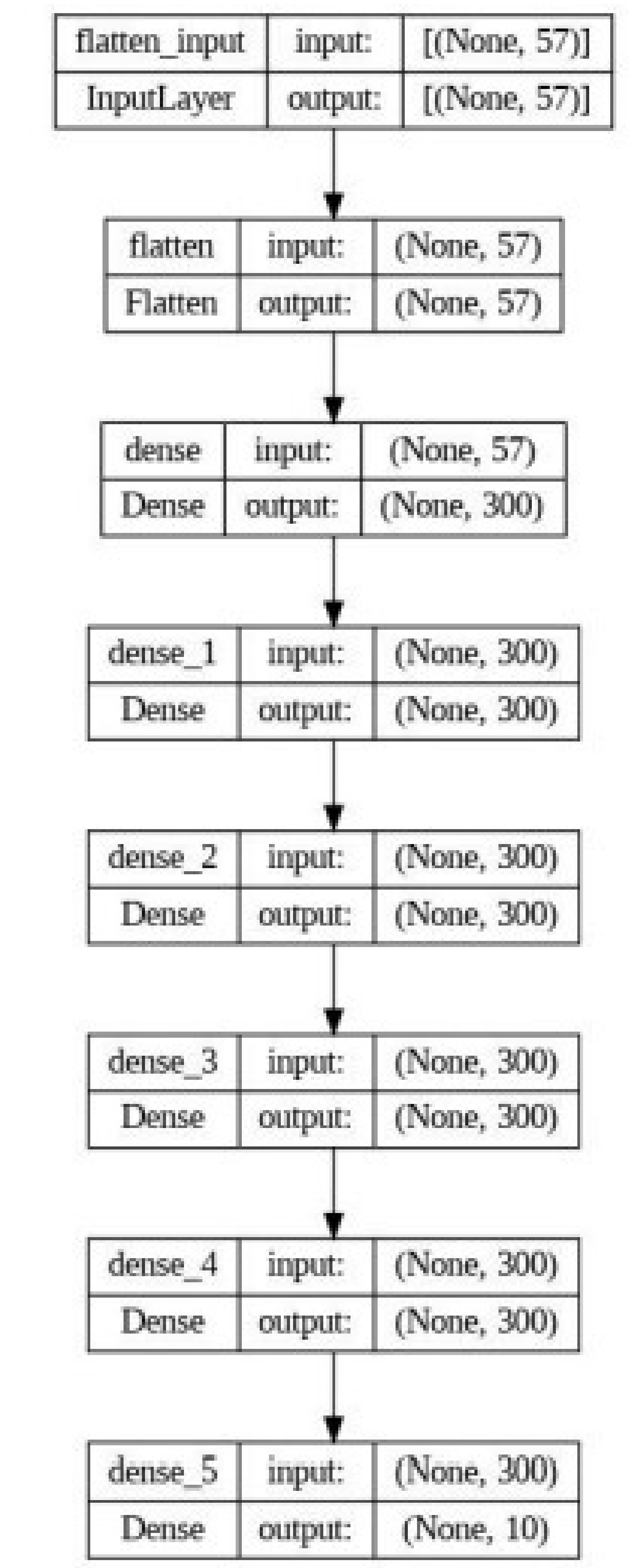
- For instance, the gamma value in the RBF and sigmoid kernels determines the extent to which a single training example impacts the model, while the degree parameter in the polynomial kernel specifies the complexity of the polynomial function used for classification. These parameters play a crucial role in fine-tuning the performance of the kernel methods.



3 NEUTRAL NETWORK

The neural network architecture consists of the following layers:

- Input Layer: A **'Flatten'** layer to convert the input data into a one-dimensional.
- Hidden Layers: Five **'Dense'** layers, activated using the ReLU function.
- Output Layer: A **'Dense'** layer with 10 neurons (corresponding to the number of music genres) and a softmax activation function.

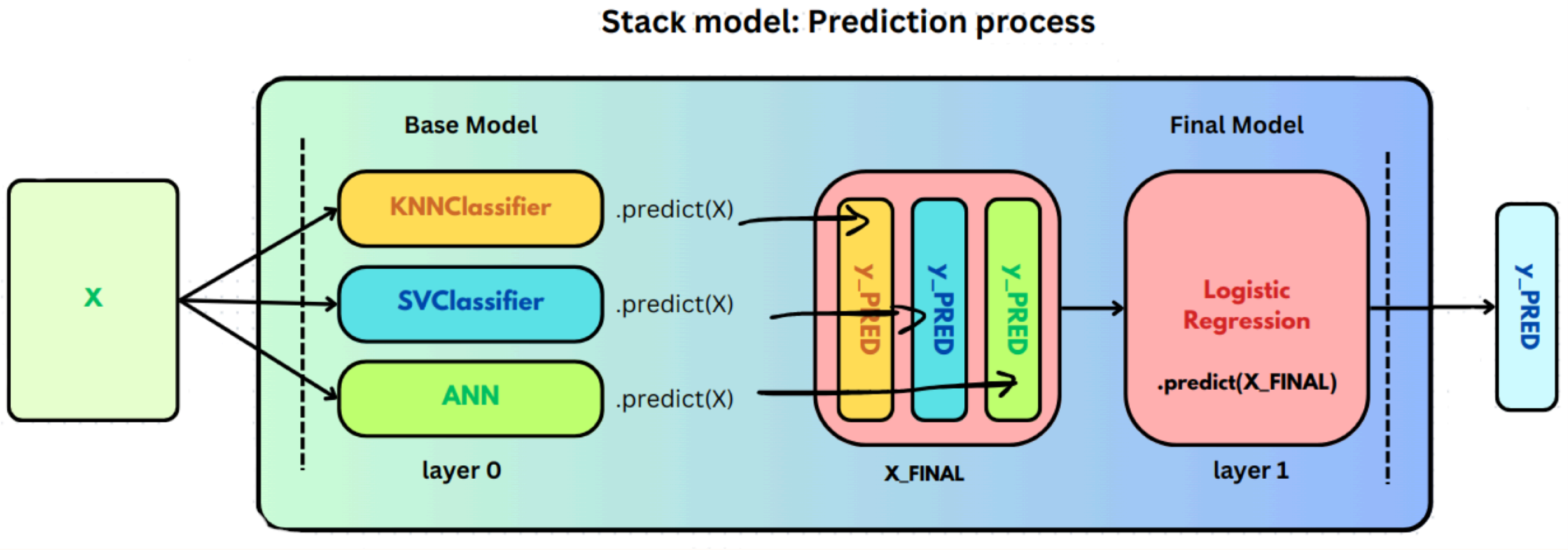


NEUTRAL NETWORK

- The network uses a variant of the gradient descent optimization algorithm named Adam with learning rate of 0.0001.
- By training on labeled data, it adjusts parameters to minimize errors and accurately predict genres of new tracks, capturing complex relationships through dense layers and activation functions.

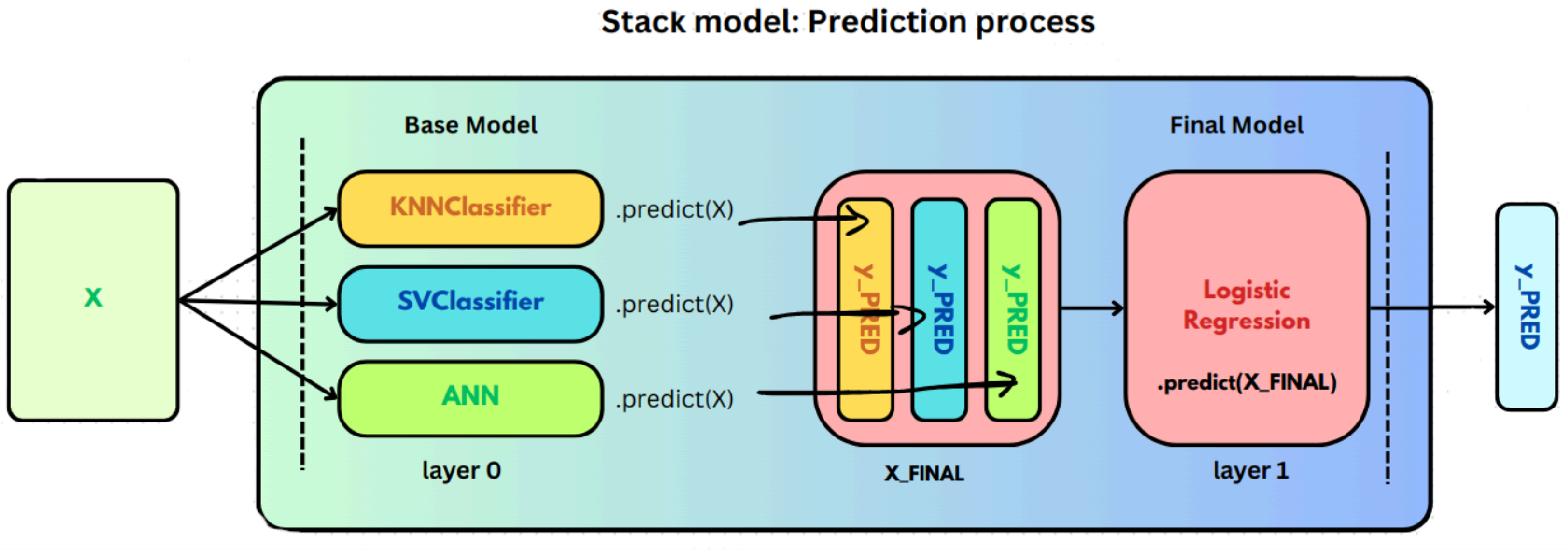
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 57)                0

 dense (Dense)               (None, 300)               17400

 dense_1 (Dense)             (None, 300)               90300

 dense_2 (Dense)             (None, 300)               90300

 dense_3 (Dense)             (None, 300)               90300

 dense_4 (Dense)             (None, 300)               90300

 dense_5 (Dense)             (None, 10)                3010

=================================================================
Total params: 381610 (1.46 MB)
Trainable params: 381610 (1.46 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

STACKING ENSEMBLE METHOD



- KNN excels in capturing local patterns
- SVM thrives in high-dimensional spaces
- While Neural Network can model complex, non-linear relationships

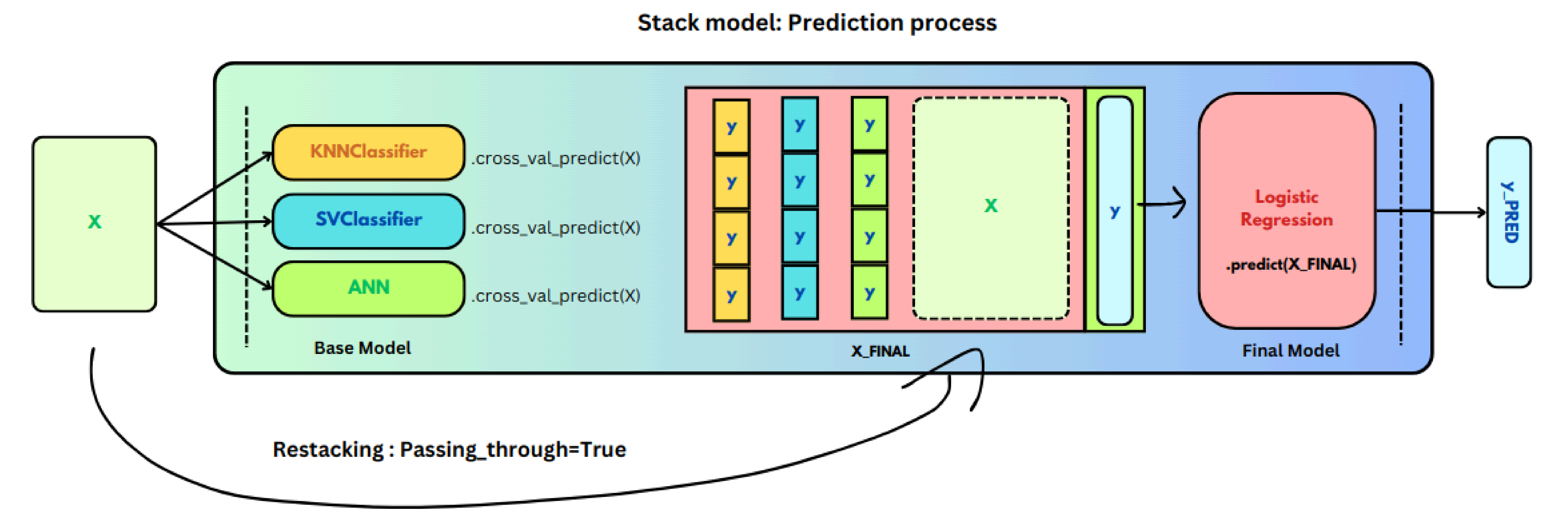
4 STACKING ENSEMBLE METHOD



→ While KNN might excel in one part of the data, SVM might perform better in another, and NN might capture intricate patterns missed by the others. If we can combine these diversity, it can helps us in covering different aspects of the data distribution, leading to a reduction in overall error. Thus, we think about an ensemble method that can combine the strengths of every models into one, and stacking technique is a perfect choice for that purpose.

4 STACKING ENSEMBLE METHOD

- How the stack fit and predict the data ?



- Individual models might suffer from high bias (underfitting) or high variance (overfitting). For instance, a high-bias model can be compensated by a low-bias but high-variance model, and vice versa.

→ Stacking helps balance these issues by averaging out the errors of individual models, just combining strengths of individual models.

EVALUATION

EVALUATE MODELS BY METRICS

	Accuracy	Precision	Recall	F1
K-Nearest-Neighbor	0.80	0.83	0.81	0.70
Support Vector Machine	0.70	0.71	0.70	0.70
Neural Network	0.80	0.79	0.79	0.79
Stacking Ensemble	0.84	0.85	0.85	0.85

- Despite its simplicity and fast speed, the K-Nearest Neighbor (KNN) algorithm delivers remarkably good performance. Its ease of implementation and efficiency contribute to its high accuracy and balanced precision and recall.

- While the Support Vector Machine (SVM) is a powerful method, it may not be the best choice in this context.

EVALUATE MODELS BY METRICS

	Accuracy	Precision	Recall	F1
K-Nearest-Neighbor	0.80	0.83	0.81	0.70
Support Vector Machine	0.70	0.71	0.70	0.70
Neural Network	0.80	0.79	0.79	0.79
Stacking Ensemble	0.84	0.85	0.85	0.85

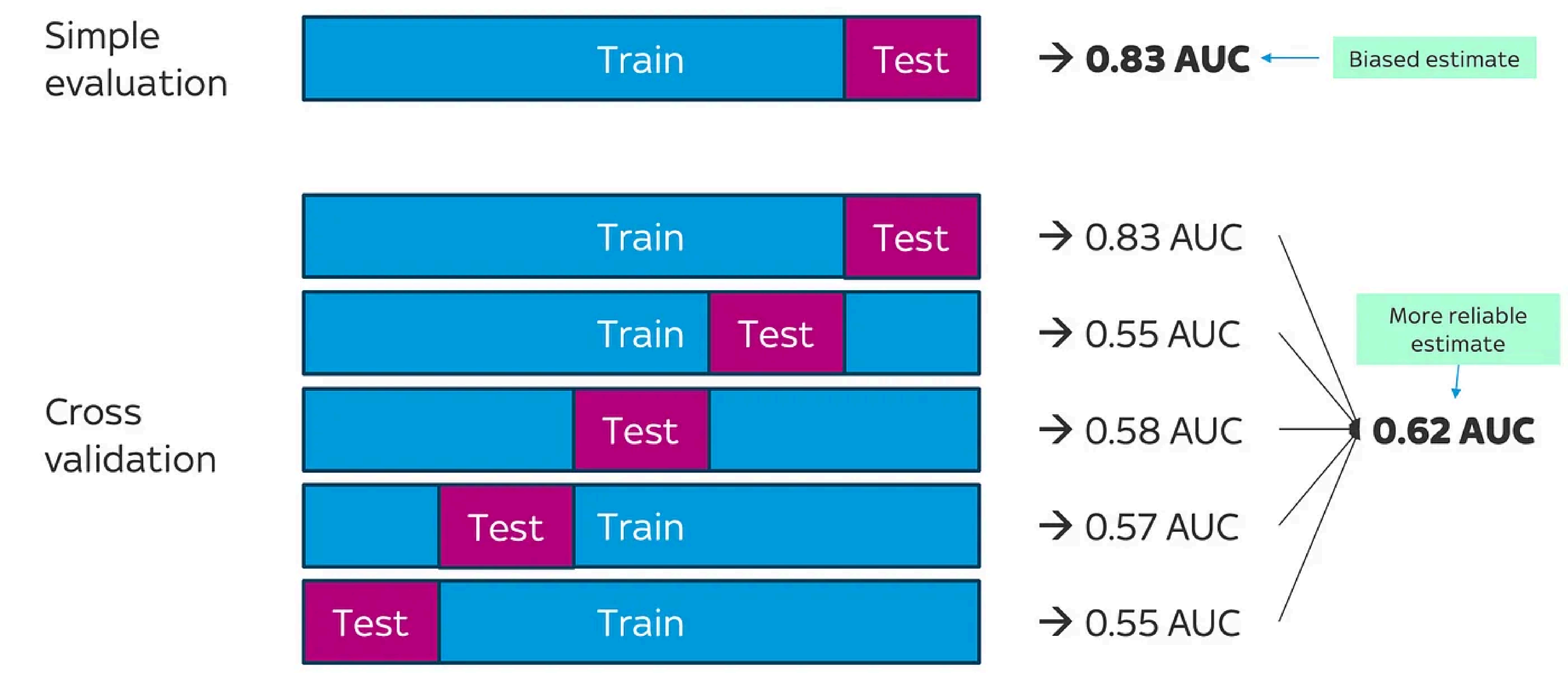
- Neural Network exhibits strong performance across various metrics, although we just tested for a random architechture and didn't tune it yet.

- The Stacking Ensemble method stands out with the highest performance metrics across the board, showcasing its potential in combining the strengths of multiple models such as KNN, SVM, Neural Network, and a final model (Logistic Regression in this case).

POSSIBLE OVERFITTING AND CROSS VALIDATION

- When we train a model on one dataset and test it on another, it might not represent real-world performance well. A single train-test split can cause overfitting, meaning the model performs well on the test data but poorly on new data. This can give us a false sense of the model's effectiveness.

→ Cross Validation as the rescue



POSSIBLE OVERFITTING AND CROSS VALIDATION

- A way to reduce the risk of overfitting, as the model is trained and validated on different subsets of the data multiple times
- Cross-validation provides a more comprehensive measure of model accuracy by averaging the results over multiple folds, which accounts for variability in the data and ensures that the model is reliable.
- Maximizes the use of the available data, by allowing every data point to be used for both training and validation purposes.

EXHAUSTIVE SEARCH FOR THE BEST MODEL WITH THE BEST CROSS VALIDATION SCORE

In the context of cross-validation, GridSearchCV from sklearn library is a powerful tool used to perform an exhaustive search over a specified parameter grid for a given model, combining cross-validation with hyperparameter tuning. By systematically working through multiple combinations of parameter values, GridSearchCV helps identify the optimal set of parameters that yield the best performance for the model.

- KNN: {n_neighbors: [1,8], weights:['uniform','distance'], p:[1,2,3]} → best config {n_neighbors=2, weights='distance'}

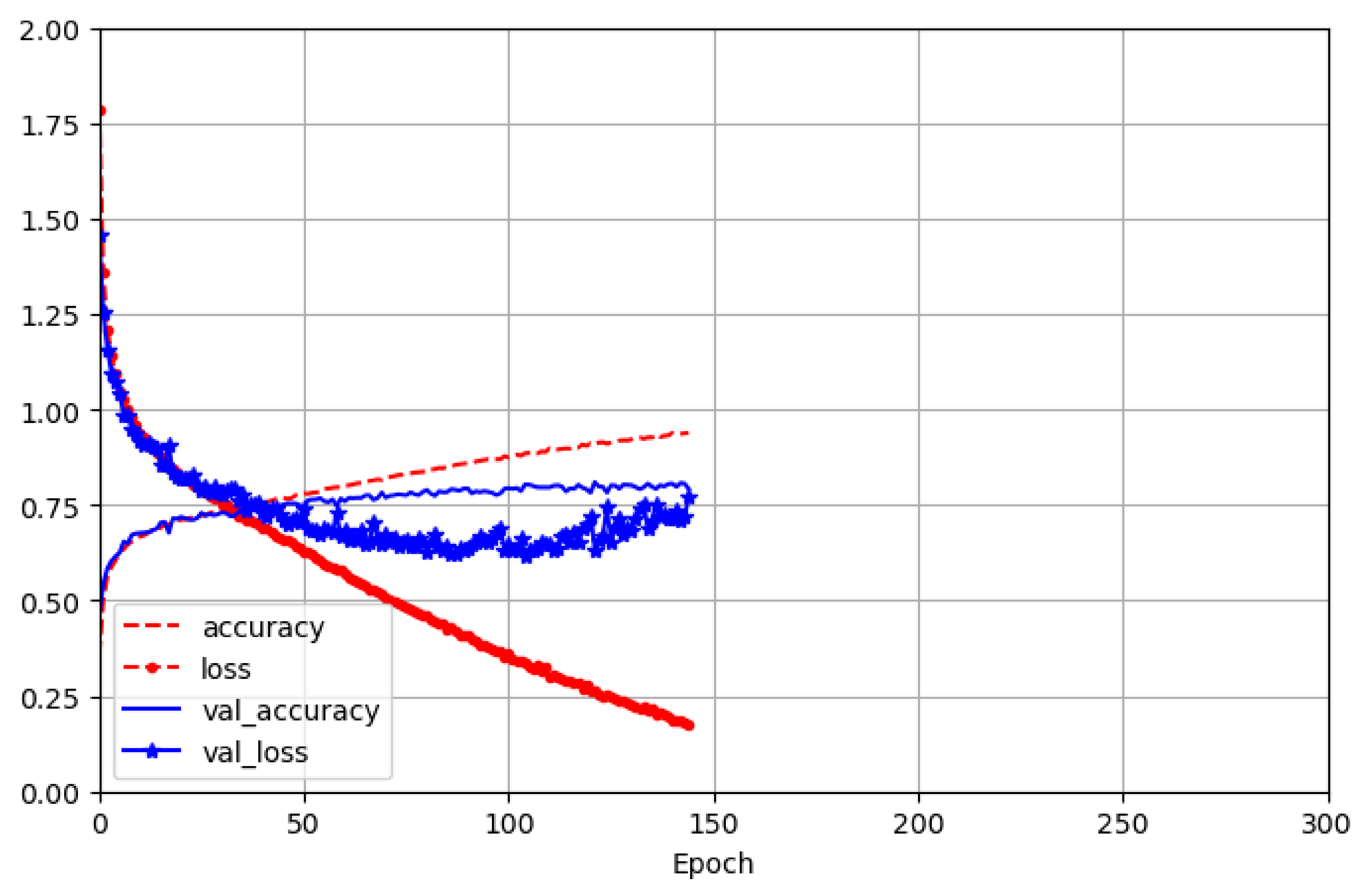
- SVM: {C: [0.1, 1, 10, 100, 1000], kernel: ['poly', 'rbf', 'sigmoid'], gamma: ['scale', 'auto']} → best config {C=100, kernel='rbf', gamma='scale'

EARLY STOPPING FOR ARTIFICIAL NEURAL NETWORK MODEL

- GridSearchCV with cross-validation is effective for optimizing KNN and SVC hyperparameters but is too computationally intensive for complex models like neural networks.

- Early stopping in Keras prevents overfitting by halting training when validation performance stops improving, monitored through validation loss.

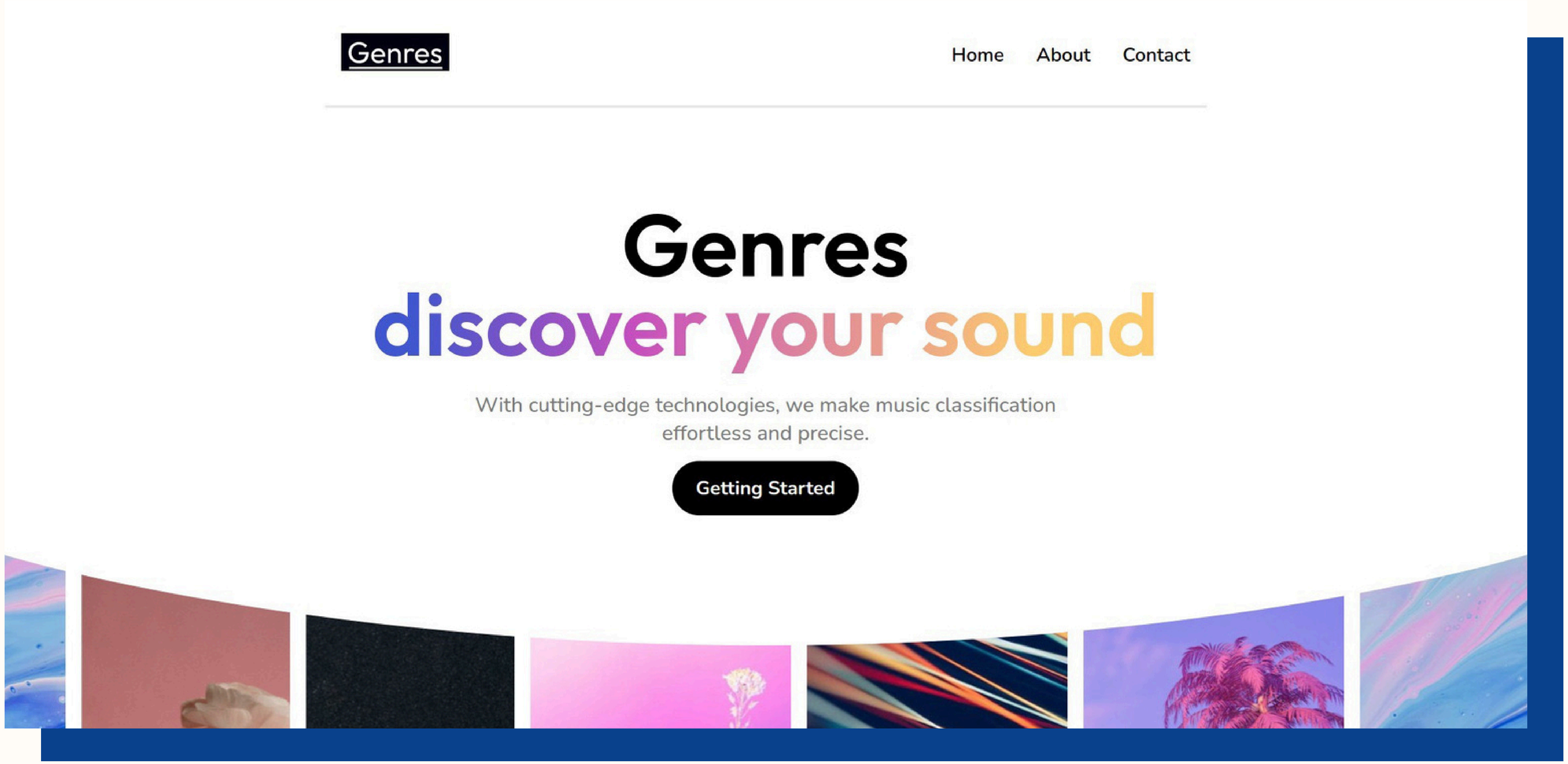
- This method saves computational resources by stopping training dynamically, avoiding unnecessary iterations.



MODEL PERFORMANCE AFTER HYPER-PARAMETER TUNING

Models	Accuracy	Precision	Recall	F1
K-Nearest-Neighbor	0.85	0.84	0.85	0.84
Support Vector Machine				
Neural Network				
Stacking Ensemble	0.88	0.88	0.88	0.88

DEMO


Genres
Home
About
Contact
Genres
discover your sound
With cutting-edge technologies, we make music classification effortless and precise.
Getting Started

THANK YOU