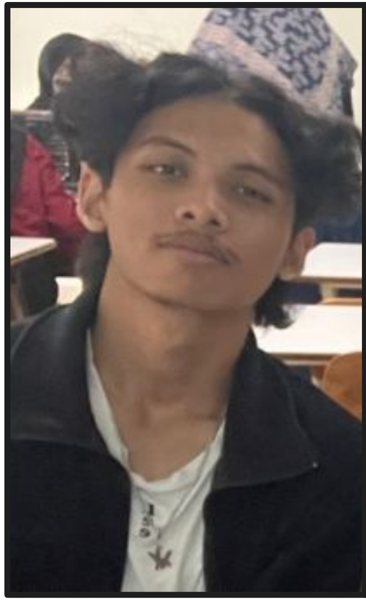
SVM Implementation Evaluation

Group 4 - Machine Learning CSB

The Dream Team



Slide Making

Satwika Nino Wandhana
23/516202/PA/22066

Slide Making

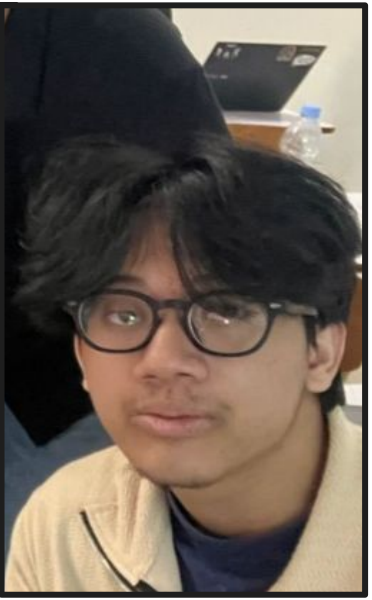
Adam Maulana Haq
23/511623/PA/21832

Coding

Daniel Winston Mandela Tulung
23/516260/PA/22080

Coding

Rindra Adriansyah
23/511559/PA/21820



Coding

Mohammad Azka Khairur Rohman
23/511608/PA/21830

Notebook & Dataset

Notebook	Dataset (Kaggle)
Google Collab	heart.csv

Single Model Experiment

```python
# Define SVM kernels
kernels = ['linear', 'poly', 'rbf', 'sigmoid']
results = {}

for kernel in kernels:
    print(f"Training SVM with {kernel} kernel...")

    if kernel == 'poly':
        model = SVC(kernel=kernel, random_state=42, degree=3)
    else:
        model = SVC(kernel=kernel, random_state=42)

    # Train the model
    model.fit(X_train, y_train)

    # Predict on the test data
    y_pred = model.predict(X_test)

    # Evaluate the model
    accuracy = accuracy_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)

    # Save a deep copy of the trained model along with results
    results[kernel] = {
        'model': copy.deepcopy(model),
        'accuracy': accuracy,
        'report': report
    }
```

Model Type	Accuracy	Precision	Sensitivity	F1
Linear	0.861	0.865	0.855	0.857
Poly (degree=3)	0.666	0.672	0.646	0.639
RBF	0.673	0.694	0.651	0.640
Sigmoid	0.541	0.271	0.496	0.350

Key Takeaway

The **linear kernel is the best single model** with ~86% accuracy and balanced metrics. Sigmoid performs poorly and is the most unreliable.

Ensemble Model Experiment

```python
# Base models (already defined earlier)
estimators = [
    ('lr', LogisticRegression(random_state=42, max_iter=1000)),
    ('dt', DecisionTreeClassifier(random_state=42)),
    ('nb', GaussianNB())
]

stacking_results = {}

for kernel in results:
    meta_model = results[kernel]['model']

    # Create stacking classifier
    stacking_clf = StackingClassifier(estimators=estimators,
                                      final_estimator=meta_model, cv=5)

    # Train
    stacking_clf.fit(X_train, y_train)

    # Predict
    y_pred_stacked = stacking_clf.predict(X_test)

    # Evaluate
    accuracy = accuracy_score(y_test, y_pred_stacked)
    report = classification_report(y_test, y_pred_stacked)

    # Save results
    stacking_results[kernel] = {
        'model': stacking_clf,
        'accuracy': accuracy,
        'report': report
    }
```

Meta Model	Accuracy	Precision	Sensitivity	F1
Linear	0.858	0.857	0.855	0.855
Poly (degree=3)	0.874	0.874	0.875	0.873
RBF	0.868	0.867	0.867	0.865
Sigmoid	0.782	0.844	0.761	0.755

Key Takeaway

The **polynomial kernel meta model outperforms others** with the highest accuracy, this is likely due to its **ability to capture complex, nonlinear patterns**. Other kernels may have underperformed due to either oversimplifying (linear) or failing to generalize well (sigmoid).

Results & Comparison

Best Performing Ensemble Model

Logistic Regression

Decision Tree Classifier

Gaussian Naive Bayes

Polynomial Kernel

Accuracy	Sensitivity	F1
0.874	0.875	0.873

Best Performing Single Model

Linear Kernel

Accuracy	Sensitivity	F1
0.861	0.855	0.857

Conclusion

The ensemble model with a polynomial kernel outperformed the best single model. This shows **ensembling improves performance by combining model strengths**.

Thank You