Tucil 1 IF3270

- Jusuf Junior Athala 13519174
- Muhammad Rifat Abiwardani 13519205

Pembagian Tugas :

1. Jusuf Junior Athala 13519174
    A. DecisionTreeClassifier
    B. Id3Estimator
    C. K Means
2. Muhammad Rifat Abiwardani 13519205
    A. LogisticRegression
    B. Neural_network
    C. SVM

In [1]:
```
from sklearn.datasets import load_breast_cancer

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score

from sklearn import tree
import six
import sys
sys.modules['sklearn.externals.six'] = six
#pip install decision-tree-id3
from id3 import Id3Estimator
from sklearn.linear_model import LogisticRegression
from sklearn.neural_network import MLPClassifier
from sklearn.svm import SVC

import matplotlib.pyplot as plt

import pandas as pd
```

In [2]:
```
# Set random state for project
RD_STATE = 1

# Load breast cancer dataset
bc_data = load_breast_cancer()

# Pisahkan menjadi training set (80%) dan testing set (20%)
bc_X_train, bc_X_test, bc_y_train, bc_y_test = train_test_split(bc_data.data, bc_data.target, test_size=0.2, random_state=RD_STATE)
```

In [3]:
```
# Load play tennis dataset
pt_df_raw = pd.read_csv("play_tennis.csv")
pt_df = pt_df_raw.copy()

# Label encoding
le = LabelEncoder()
pt_classes = {}
for column in pt_df.columns:
    le.fit(pt_df[column])
```

```
    pt_df[column] = le.transform(pt_df[column])
    pt_classes[column] = le.classes_
pt_X_train, pt_X_test, pt_y_train, pt_y_test = train_test_split(pt_df[['outlook', 'temp', 'humidity', 'wind']], pt_df[['play']], test_size=0.2, random_state=RD_STATE)
```

In [4]:

```
from sklearn.tree import export_text as ex_t

# DecisionTreeClassifier

# Fit classifier
bc_dtc_clf = tree.DecisionTreeClassifier().fit(bc_X_train, bc_y_train)

pt_dtc_clf = tree.DecisionTreeClassifier().fit(pt_X_train, pt_y_train)

# Print accuracy dan nilai f1
bc_fn = bc_data.feature_names
bc_t = ex_t(bc_dtc_clf, feature_names=list(bc_fn))
pt_t = ex_t(pt_dtc_clf, feature_names=list(['Outlook','Temprature','Humidity','Wind'] ))

print("DecisionTreeClassifier:")
print("• Breast cancer dataset")
print("\tAccuracy: ", bc_dtc_clf.score(bc_X_test, bc_y_test))
print("\tf1_score: ", f1_score(bc_y_test, bc_dtc_clf.predict(bc_X_test)))
print()
print(bc_t)
print("• Play tennis dataset")
print("\tAccuracy: ", pt_dtc_clf.score(pt_X_test, pt_y_test['play']))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_dtc_clf.predict(pt_X_test)))
print()
print(pt_t)
```

```
DecisionTreeClassifier:
• Breast cancer dataset
        Accuracy:  0.9473684210526315
        f1_score:  0.9589041095890412

|--- worst perimeter <= 106.05
|   |--- worst concave points <= 0.16
|   |   |--- worst concave points <= 0.14
|   |   |   |--- area error <= 48.98
|   |   |   |   |--- class: 1
|   |   |   |--- area error >  48.98
|   |   |   |   |--- worst compactness <= 0.08
|   |   |   |   |   |--- class: 0
|   |   |   |   |--- worst compactness >  0.08
|   |   |   |   |   |--- class: 1
|   |   |--- worst concave points >  0.14
|   |   |   |--- mean texture <= 20.84
|   |   |   |   |--- class: 1
|   |   |   |--- mean texture >  20.84
|   |   |   |   |--- class: 0
|   |--- worst concave points >  0.16
|   |   |--- mean texture <= 16.22
|   |   |   |--- class: 1
|   |   |--- mean texture >  16.22
|   |   |   |--- class: 0
|--- worst perimeter >  106.05
|   |--- worst texture <= 20.65
|   |   |--- worst perimeter <= 116.80
|   |   |   |--- class: 1
|   |   |--- worst perimeter >  116.80
|   |   |   |--- smoothness error <= 0.00
|   |   |   |   |--- class: 1
|   |   |   |--- smoothness error >  0.00
|   |   |   |   |--- class: 0
|   |--- worst texture >  20.65
```

```
|   |   |--- mean concave points <= 0.05
|   |   |   |--- concave points error <= 0.01
|   |   |   |   |--- class: 0
|   |   |   |--- concave points error >  0.01
|   |   |   |   |--- class: 1
|   |   |--- mean concave points >  0.05
|   |   |   |--- mean smoothness <= 0.08
|   |   |   |   |--- class: 1
|   |   |   |--- mean smoothness >  0.08
|   |   |   |   |--- class: 0
```

```
• Play tennis dataset
       Accuracy:  0.6666666666666666
       f1_score:  0.6666666666666666
```

```
|--- Outlook <= 0.50
|   |--- class: 1
|--- Outlook >  0.50
|   |--- Humidity <= 0.50
|   |   |--- class: 0
|   |--- Humidity >  0.50
|   |   |--- Wind <= 0.50
|   |   |   |--- Outlook <= 1.50
|   |   |   |   |--- class: 0
|   |   |   |--- Outlook >  1.50
|   |   |   |   |--- class: 1
|   |   |--- Wind >  0.50
|   |   |   |--- class: 1
```

Decision Tree adalah salah satu metode untuk memprediksi target data dengan membuat model pohon yang dibuat berdasarkan pembelajaran nilai pada data training. Sckit menggunakan algoritma CART (Classification and Regression Trees) untuk DecisionTreeClassifier. Algoritma ini mengelompokan setiap pola atau data dalam sebuah kelas-kelas tertentu dan memodelkannya dalam bentuk pohon.

• Breast cancer dataset
• Accuracy: 0.9473684210526315
Hasil penilaian berdasarkan accuracy menunjukkan 94% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.
• f1_score: 0.9589041095890412
Hasil penilaian berdasarkan F1 menunjukkan 95% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.
Berdasarkan hasil, penilaian berdasarkan F1 memiliki nilai yang lebih besar dibandingkan dengan accuracy.

• Play tennis dataset
• Accuracy: 0.6666666666666666
Hasil penilaian berdasarkan accuracy menunjukkan 66% ketepatan prediksi. Hasil ini merupakan hasil yang cukup rendah.
• f1_score: 0.6666666666666666
Hasil penilaian berdasarkan F1 menunjukkan 66% ketepatan prediksi. Hasil ini merupakan hasil yang cukup rendah.
Salah satu penyebab dari penilaian rendah untuk accuracy dan F1 adalah data test untuk dataset Play-Tennis yang berjumlah sedikit.

In [5]:

```
from id3 import export_text
# Id3Estimator

# Fit classifier
bc_id3_clf = Id3Estimator().fit(bc_X_train, bc_y_train)

pt_id3_clf = Id3Estimator().fit(pt_X_train, pt_y_train['play'])

# Print accuracy dan nilai f1
print("DecisionTreeClassifier:")
print("• Breast cancer dataset")
print("\tAccuracy: ", accuracy_score(bc_y_test, bc_id3_clf.predict(bc_X_test)))
print("\tf1_score: ", f1_score(bc_y_test, bc_id3_clf.predict(bc_X_test)))
print(export_text(bc_id3_clf.tree_, bc_fn))
```

```
print("• Play tennis dataset")
print("\tAccuracy: ", accuracy_score(pt_y_test['play'], pt_id3_clf.predict(pt_X_test)))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_id3_clf.predict(pt_X_test)))
print(export_text(pt_id3_clf.tree_, feature_names=['Outlook','Temprature','Humidity','Wind'] ))
```

```
DecisionTreeClassifier:
• Breast cancer dataset
        Accuracy:  0.9385964912280702
        f1_score:  0.953020134228188

worst perimeter <=105.15
|   worst concave points <=0.14
|   |   radius error <=0.64: 1 (249)
|   |   radius error >0.64
|   |   |   mean radius <=12.27: 0 (1)
|   |   |   mean radius >12.27: 1 (2)
|   worst concave points >0.14
|   |   worst texture <=25.94: 1 (6)
|   |   worst texture >25.94
|   |   |   mean compactness <=0.12
|   |   |   |   mean radius <=14.06: 1 (2)
|   |   |   |   mean radius >14.06: 0 (1)
|   |   |   mean compactness >0.12: 0 (5)
worst perimeter >105.15
|   worst concave points <=0.15
|   |   worst texture <=19.91: 1 (13)
|   |   worst texture >19.91
|   |   |   worst radius <=16.80
|   |   |   |   mean smoothness <=0.09: 1 (8)
|   |   |   |   mean smoothness >0.09
|   |   |   |   |   smoothness error <=0.00: 1 (2)
|   |   |   |   |   smoothness error >0.00: 0 (5)
|   |   |   worst radius >16.80
|   |   |   |   worst concavity <=0.21
|   |   |   |   |   mean texture <=21.26: 1 (2)
|   |   |   |   |   mean texture >21.26: 0 (2)
|   |   |   |   worst concavity >0.21: 0 (21)
|   worst concave points >0.15
|   |   mean texture <=15.35
|   |   |   mean radius <=14.89: 1 (1)
|   |   |   mean radius >14.89: 0 (3)
|   |   mean texture >15.35: 0 (132)

• Play tennis dataset
        Accuracy:  0.6666666666666666
        f1_score:  0.6666666666666666

Outlook <=0.50: 1 (3)
Outlook >0.50
|   Humidity <=0.50: 0 (3)
|   Humidity >0.50
|   |   Wind <=0.50
|   |   |   Temprature <=1.00: 0 (1)
|   |   |   Temprature >1.00: 1 (1)
|   |   Wind >0.50: 1 (3)
```

Id3Estimator adalah salah satu algoritma decision tree. Algoritma ini melakukan pencarian secara menyeluruh (greedy) pada semua kemungkinan pohon keputusan

• Breast cancer dataset

• Accuracy: 0.9385964912280702

Hasil penilaian berdasarkan accuracy menunjukkan 93% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.

• f1_score: 0.953020134228188

Hasil penilaian berdasarkan F1 menunjukkan 95% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.

Berdasarkan hasil, penilaian berdasarkan F1 memiliki nilai yang lebih besar dibandingkan dengan accuracy.

• Play tennis dataset

• Accuracy: 0.6666666666666666

Hasil penilaian berdasarkan accuracy menunjukkan 66% ketepatan prediksi. Hasil ini merupakan hasil yang cukup rendah.

• f1_score: 0.6666666666666666

Hasil penilaian berdasarkan F1 menunjukkan 66% ketepatan prediksi. Hasil ini merupakan hasil yang cukup rendah.

Salah satu penyebab dari penilaian rendah untuk accuracy dan F1 adalah data test untuk dataset Play-Tennis yang berjumlah sedikit.

In [6]:

```
# K Means
from sklearn.cluster import KMeans
# Fit classifier
bc_km_clf = KMeans(random_state=RD_STATE,n_clusters=2, max_iter=5).fit(bc_X_train, bc_y_train)

pt_km_clf = KMeans(random_state=RD_STATE,n_clusters=2, max_iter=300).fit(pt_X_train, pt_y_train)

# Print accuracy dan nilai f1

print("DecisionTreeClassifier:")
print("• Breast cancer dataset")
print("\tAccuracy: ", accuracy_score(bc_y_test, bc_km_clf.predict(bc_X_test)))
print("\tf1_score: ", f1_score(bc_y_test, bc_km_clf.predict(bc_X_test)))

print("• Play tennis dataset")
print("\tAccuracy: ", accuracy_score(pt_y_test['play'], pt_km_clf.predict(pt_X_test)))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_km_clf.predict(pt_X_test)))
```

```
DecisionTreeClassifier:
• Breast cancer dataset
        Accuracy:  0.8070175438596491
        f1_score:  0.8674698795180723
• Play tennis dataset
        Accuracy:  0.0
        f1_score:  0.0
```

K Means adalah algoritma dengan mempartisi data yang ada kedalam satu atau lebih cluster atau kelompok data berdasarkan karakteristiknya. Data yang memiliki karakteristik berbeda akan dikelompokan kedalam cluster yang lain. Pada setiap cluster terdapat titik pusat (centroid) yang merepresentasikan cluster tersebut.

• Breast cancer dataset

• Accuracy: 0.8070175438596491

Hasil penilaian berdasarkan accuracy menunjukkan 80% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.

• f1_score: 0.8674698795180723

Hasil penilaian berdasarkan F1 menunjukkan 86% ketepatan prediksi. Hasil ini merupakan hasil yang cukup tinggi.

Berdasarkan hasil, penilaian berdasarkan F1 memiliki nilai yang lebih besar dibandingkan dengan accuracy.

• Play tennis dataset

• Accuracy: 0.0

Hasil penilaian berdasarkan accuracy menunjukkan 0% ketepatan prediksi. Hasil ini merupakan hasil yang rendah.

• f1_score: 0.0

Hasil penilaian berdasarkan F1 menunjukkan 0% ketepatan prediksi. Hasil ini merupakan hasil yang rendah.

Salah satu penyebab dari penilaian rendah untuk accuracy dan F1 adalah data test untuk dataset Play-Tennis yang berjumlah sedikit.

In [7]:

```
# Logistic Regression

# Data preprocessing: normalisasi data agar setiap feature memiliki orde magnituda yang sama
bc_scaler = StandardScaler().fit(bc_data.data)
bc_X_train_scaled = bc_scaler.transform(bc_X_train)
bc_X_test_scaled = bc_scaler.transform(bc_X_test)
```

```
pt_scaler = StandardScaler().fit(pt_df[['outlook', 'temp', 'humidity', 'wind']])
pt_X_train_scaled = pt_scaler.transform(pt_X_train)
pt_X_test_scaled = pt_scaler.transform(pt_X_test)

# Fit classifier
bc_lr_clf = LogisticRegression(random_state=RD_STATE).fit(bc_X_train_scaled, bc_y_train)

pt_lr_clf = LogisticRegression(random_state=RD_STATE).fit(pt_X_train_scaled, pt_y_train['play'])

# Print accuracy dan nilai f1
print("Logistic Regression Performance:")
print("• Breast cancer dataset")
print("\tAccuracy: ", bc_lr_clf.score(bc_X_test_scaled, bc_y_test))
print("\tf1_score: ", f1_score(bc_y_test, bc_lr_clf.predict(bc_X_test_scaled)))
print("• Play tennis dataset")
print("\tAccuracy: ", pt_lr_clf.score(pt_X_test_scaled, pt_y_test['play']))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_lr_clf.predict(pt_X_test_scaled)))
```

```
Logistic Regression Performance:
• Breast cancer dataset
        Accuracy:  0.9736842105263158
        f1_score:  0.9793103448275863
• Play tennis dataset
        Accuracy:  0.6666666666666666
        f1_score:  0.8
```

Logistic regression untuk dua kelas melibatkan fungsi logit yang menggunakan nilai dot product dari vektor bobot b dengan vektor inputs x ditambah bias sebagai parameter exponennya. Dalam bentuk rumus:

• $y_{pred} = 1/(1+\exp(-(\sum(b_i.x_i)+b_0)))$

Training loop pada model logistic regression merupakan proses perhitungan cost dari hasil prediksi terhadap label sebetulnya, kemudian memperbarui nilai vektor bobot b, misal dengan stochastic gradient descent menggunakan rumus $b_j = b_j+a*(y_i-p_i)*x_{ij}$, sehingga prediksi training set converge terhadap nilai asli

Kinerja logistic regression pada dataset breast cancer cukup baik, accuracy 97% dan nilai f1 98%. Sementara itu, play tennis dataset memiliki accuracy 66.7% yang cukup buruk dan nilai f1 80% yang rata-rata. Namun ini karena ukuran test set play tennis cukup kecil, hanya 3, dan training set yang relatif kecil, hanya 11. Sebagai perbandingan, dataset breast cancer memiliki ukuran training set 455 dan test set 114. Oleh karena itu, terdapat noise yang mengganggu hasil analisis kinerja. Training set kecil menyebabkan kemungkinan overfit meningkat, dan testing set kecil berarti variance dataset condong lebih besar. Jika dataset lebih besar, maka accuracy classifier pada dataset play tennis kemungkinan berubah menjadi tidak lagi 66.7%.

In [8]:

```
# Neural Network: Multi-Layer Perceptron Model

# Fit classifier
bc_mlp_clf = MLPClassifier(random_state=RD_STATE, activation='relu', max_iter=1000, hidden_layer_sizes=(100,)).fit(bc_X_train, bc_y_train)
1
pt_mlp_clf = MLPClassifier(random_state=RD_STATE, activation='relu', max_iter=1000, hidden_layer_sizes=(100,)).fit(pt_X_train, pt_y_train['play'])

# Print accuracy dan nilai f1
print("Neural Network Performance:")
print("• Breast cancer dataset")
print("\tAccuracy: ", bc_mlp_clf.score(bc_X_test, bc_y_test))
print("\tf1_score: ", f1_score(bc_y_test, bc_mlp_clf.predict(bc_X_test)))
print("• Play tennis dataset")
print("\tAccuracy: ", pt_mlp_clf.score(pt_X_test, pt_y_test['play']))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_mlp_clf.predict(pt_X_test)))
```

```
Neural Network Performance:
• Breast cancer dataset
        Accuracy:  0.9473684210526315
        f1_score:  0.9594594594594595
• Play tennis dataset
        Accuracy:  1.0
        f1_score:  1.0
```

In [9]:
```
bc_mlp_params = bc_mlp_clf.get_params()
pt_mlp_params = pt_mlp_clf.get_params()

print("Parameter MLP dataset breast cancer:")
print("• Activation function\t:", bc_mlp_params['activation'])
print("• Learning rate\t\t:", bc_mlp_params['learning_rate_init'])
print("• Max iterasi\t\t:", bc_mlp_params['max_iter'])
print("• Jumlah hidden layer\t:", bc_mlp_params['hidden_layer_sizes'][0])
print("\nParameter MLP dataset play tennis:")
print("• Activation function\t:", pt_mlp_params['activation'])
print("• Learning rate\t\t:", pt_mlp_params['learning_rate_init'])
print("• Max iterasi\t\t:", pt_mlp_params['max_iter'])
print("• Jumlah hidden layer\t:", pt_mlp_params['hidden_layer_sizes'][0])
```

```
Parameter MLP dataset breast cancer:
• Activation function   : relu
• Learning rate         : 0.001
• Max iterasi           : 1000
• Jumlah hidden layer   : 100

Parameter MLP dataset play tennis:
• Activation function   : relu
• Learning rate         : 0.001
• Max iterasi           : 1000
• Jumlah hidden layer   : 100
```

Multilayer Perceptron merupakan jenis Artificial Neural Network, sebuah model machine learning yang menggunakan node perceptron sebagai blok pembangun modelnya. Dalam MLP, terdapat tiga jenis layer node: input layer, hidden layer, dan output layer. Input layer berjumlah sebanyak parameter input, kemudian nilai setiap node pada layer berikutnya dipengaruhi oleh node-node input berdasarkan bobot tertentu. Untuk suatu node yang dipengaruhi oleh nilai node pada layer sebelumnya x_{ij}, nilai terbobotnya merupakan $\sum(\sum(w_{ij}*x_{ij}) + b_i)$. Nilai ini kemudian dilalui sebuah fungsi aktivasi, yang digunakan untuk memetakan nilai terbobot ke suatu rentang tertentu. Hasil prediksi dari output layer digunakan untuk menghitung cost dan memperbarui nilai bobot layer terakhir sebelum output layer. Nilai baru pada layer tersebut digunakan kembali untuk menghitung cost dan memberbarui nilai bobot layer sebelumnya, dan seterusnya hingga sampai ke layer pertama.

Terdapat beberapa jenis fungsi aktivasi seperti fungsi sigmoid (seperti yang digunakan di logistic regression), fungsi hyperbolic tan, ReLU, dan fungsi identitas. Fungsi sigmoid memiliki bentuk S horizontal yang memetakan nilai terbobot menjadi 0 atau 1. Fungsi tanh memetakan nilai terbobot menjadi -1 atau 1, kemudian fungsi ReLU memetakan nilai negatif menjadi 0 dan nilai positif tidak diubah, sementara fungsi identitas tidak mengubah nilai terbobot.

Dalam tugas ini, fungsi ReLU digunakan. Fungsi ReLU cukup umum untuk digunakan dalam MLP karena menghindari masalah numerik yang dialami fungsi sigmoid dan tanh saat melakukan gradient descent. Berikut rumus fungsi ReLU:

• $x_{ij} = \max(0, x_{ij})$

Kinerja MLP pada dataset breast cancer dan play tennis cukup baik, dengan accuracy 94.7% dan nilai f1 95.9% pada dataset breast cancer serta accuracy 100% dan nilai f1 100% pada dataset play tennis. Namun karena dataset play tennis cukup kecil, maka terdapat kemungkinan bahwa terdapat noise yang mempengaruhi kinerja classifier MLP-nya.

In [10]:
```
# Support Vector Machine: C-Support Vector Classification

# Data preprocessing: normalisasi data agar setiap feature memiliki orde magnituda yang sama
bc_scaler = StandardScaler().fit(bc_data.data)
bc_X_train_scaled = bc_scaler.transform(bc_X_train)
bc_X_test_scaled = bc_scaler.transform(bc_X_test)

pt_scaler = StandardScaler().fit(pt_df[['outlook', 'temp', 'humidity', 'wind']])
pt_X_train_scaled = pt_scaler.transform(pt_X_train)
pt_X_test_scaled = pt_scaler.transform(pt_X_test)

# Fit classifier
bc_svc_clf = SVC(random_state=RD_STATE, gamma='auto').fit(bc_X_train_scaled, bc_y_train)

pt_svc_clf = SVC(random_state=RD_STATE, gamma='auto').fit(pt_X_train_scaled, pt_y_train['play'])
```

```
# Print accuracy dan nilai f1
print("SVM Performance:")
print("• Breast cancer dataset")
print("\tAccuracy: ", bc_svc_clf.score(bc_X_test_scaled, bc_y_test))
print("\tf1_score: ", f1_score(bc_y_test, bc_svc_clf.predict(bc_X_test_scaled)))
print("• Play tennis dataset")
print("\tAccuracy: ", pt_svc_clf.score(pt_X_test_scaled, pt_y_test['play']))
print("\tf1_score: ", f1_score(pt_y_test['play'], pt_svc_clf.predict(pt_X_test_scaled)))
```

```
SVM Performance:
• Breast cancer dataset
        Accuracy:  0.9736842105263158
        f1_score:  0.9793103448275863
• Play tennis dataset
        Accuracy:  1.0
        f1_score:  1.0
```

In [11]:

```
bc_svc_params = bc_svc_clf.get_params()
pt_svc_params = pt_svc_clf.get_params()

print("Parameter MLP dataset breast cancer:")
print("• Penalty parameter\t:", bc_svc_params['C'])
print("• Kernel\t\t:", bc_svc_params['kernel'])
print("\nParameter MLP dataset play tennis:")
print("• Penalty parameter\t:", pt_svc_params['C'])
print("• Kernel\t\t:", pt_svc_params['kernel'])
```

```
Parameter MLP dataset breast cancer:
• Penalty parameter     : 1.0
• Kernel                : rbf
Parameter MLP dataset play tennis:
• Penalty parameter     : 1.0
• Kernel                : rbf
```

C-Support Vector Classification merupakan kelas SVM, sebuah model machine learning yang melakukan mapping pada data input menjadi titik data fitur berdimensi tinggi, untuk kemudian dicari batasan-batasan ruang yang dapat membagi data dengan baik. Untuk melakukan mapping, sebuah method disebut kernel digunakan. Dalam tugas ini, kernel Radial Basis Function (RBF) digunakan. Untuk dua titik data x dan x', Kernel RBF memiliki rumus sebagai berikut.

• $K(x, x') = \exp(-||x-x'||^2/(2s^2))$

Dimana $1/(2s^2)$ juga sering ditulis sebagai gamma. Parameter gamma mengendalikan fleksibilitas daerah decision boundary; jika gamma rendah, maka batasan ruang cenderung kurva sederhana, sementara jika gamma tinggi, maka batasan ruang dapat membentuk lingkaran daerah-daerah tentu yang terlihat lebih terpengaruh oleh masing data point. Perilaku kernel juga terpengaruhi oleh penalty parameter C, dimana semakin tinggi C, maka kernel semakin tidak toleran terhadap misklasifikasi.

Kinerja SVC pada dataset breast cancer cukup baik, dengan accuracy 97.3% dan nilai f1 98%. Begitu pula kinerjanya cukup baik pada dataset play tennis, dengan accuracy 100% dan nilai f1 100%. Namun sama seperti model-model sebelumnya, terdapat kemungkinan bahwa kecilnya dataset play tennis menyebabkan evaluasi kinerja classifier tidak representatif akan kinerja aslinya jika di-test pada dataset yang lebih besar.