
# Multiclass Classification Assignment 1

## Applied Machine Learning ELG5255[EG]

Group name: Group 6

Ahmed Fares Saad Eldin Khalifa

Abdallah Medhat Mohamed Rashed

# Table of Contents

# 1. Objectives

The purpose of this assignment to do multiclass classification on ("Seeds dataset") using OvR technique. Which work as the following:

- Define the target label from labels column.
- Then search for the desired label in the column and put it with value (1).
- Anything else is the label column is labeled to (-1).

At the end we have 3 binarized models for the Iris dataset. Train and evaluate them using SVC and Perceptron, and compare the two models.

# 2. Implementation

## 2.1 Import libraries for use in the task.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.linear_model import Perceptron
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, plot_confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import VotingClassifier
from sklearn import tree
import seaborn as sn
from sklearn.tree import DecisionTreeClassifier
pd.set_option('display.max_rows', None)
```

*Figure 1:Import libraries*

## 2.2 Load and import the dataset using pandas.

```
seeds_train = pd.read_csv("seeds_train.csv")
seeds_test = pd.read_csv("seeds_test.csv")
```

*Figure 2: Load dataset*

## 2.3 Data Preparation & Evaluation

- Splitting the training and testing data to features (x_seeds_train) and label (y_seeds_train) and removing the class Kama (1) from the dataset of training and testing.

```
x_seeds_train = seeds_train.iloc[57:, :-1].to_numpy()
y_seeds_train = seeds_train.iloc[57:, -1]

df_filtered = seeds_test[seeds_test['1'] !=1]


x_seeds_test = df_filtered.iloc[:,:-1].to_numpy()
y_seeds_test = df_filtered.iloc[:, -1]
```

*Figure 3: split dataset*

- Applying the machine learning algorithm (Perceptron and SVM) to the dataset, after applying both algorithms, **the accuracy of two algorithms is equal**.

```
# ---------------------------------- SVM for train data ------------------
clf= SVC(kernel = 'linear', random_state = 0)
clf.fit(x_seeds_train,y_seeds_train)
# Predicting the Test set results
y_pred=clf.predict(x_seeds_test)
#Making the Confusion Matrix and Classification Report
print("Accuracy for model_SVM: ",clf.score(x_seeds_test, y_seeds_test)*100)
print('\nClassification Report_SVM:\n')
print(classification_report(y_seeds_test, y_pred))
print('Confusion Matrix_SVM:\n')
print(confusion_matrix(y_seeds_test, y_pred))
plot_confusion_matrix(clf, x_seeds_test, y_seeds_test)
```

*Figure 4: applied SVC algorithm on dataset*

```
# ---------------------------------- Perceptron for train data -------------
clf = Perceptron(n_iter_no_change=80)
clf.fit(x_seeds_train,y_seeds_train)
y_pred=clf.predict(x_seeds_test)
#print(y_pred)
#Making the Confusion Matrix and Classification Report
print("Accuracy for model_Perceptron: ",clf.score(x_seeds_test, y_seeds_test)*
print('\nClassification Report_Perceptron:\n')
print(classification_report(y_seeds_test, y_pred))
print('Confusion Matrix_Perceptron:\n')
print(confusion_matrix(y_seeds_test, y_pred))
plot_confusion_matrix(clf, x_seeds_test, y_seeds_test)
```

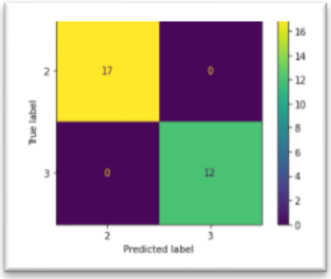
*Figure 5: applied Perceptron algorithm on dataset*

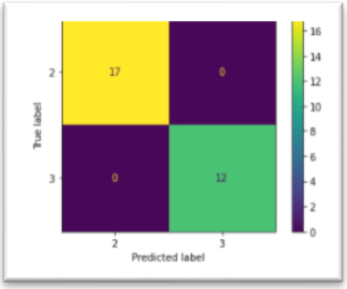
```
Accuracy for model_SVM:  100.0

Classification Report_SVM:

              precision    recall  f1-score   support

           2       1.00      1.00      1.00        17
           3       1.00      1.00      1.00        12

    accuracy                           1.00        29
   macro avg       1.00      1.00      1.00        29
weighted avg       1.00      1.00      1.00        29
```

*Figure 6:SVC Report on dataset*

```
Accuracy for model_Perceptron:  100.0

Classification Report_Perceptron:

              precision    recall  f1-score   suppor

           2       1.00      1.00      1.00        1
           3       1.00      1.00      1.00        1

    accuracy                           1.00        2
   macro avg       1.00      1.00      1.00        2
weighted avg       1.00      1.00      1.00        2
```

*Figure 7: Perceptron Report on dataset*



- Using binarized function for applying OVR (1 for positive class, -1 for negative class)

```
def decision_boundary(model,x,Labels_1,title,x_label,y_label):
    X_set, y_set = x,Labels_1
    X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                         np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
    plt.contourf(X1, X2, model.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
                 alpha = 0.20, cmap = ListedColormap(('red', 'green')))
    plt.xlim(X1.min(), X1.max())
    plt.ylim(X2.min(), X2.max())
    for i, j in enumerate(np.unique(y_set)):
        plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                    color = ListedColormap(('red', 'green'))(i), label = j)
    plt.title(title)
    plt.xlabel(x_label)
    plt.ylabel(y_label)
    plt.legend()
    plt.show()
```

*Figure 8: binarized function*

- Applying SVM with OVR (bainarized_labels) with each class in addition to obtaining confusion matrix, accuracy and plotting the decision boundary.

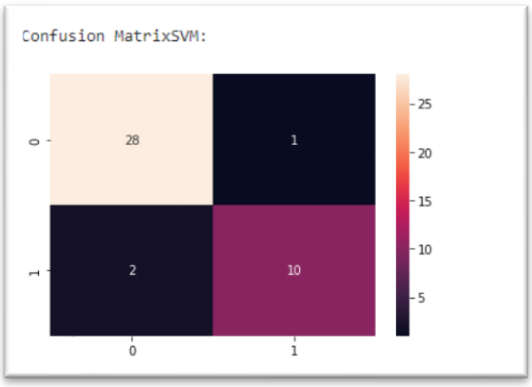



*Figure 9: confusion matrix on SVM model with OVR*

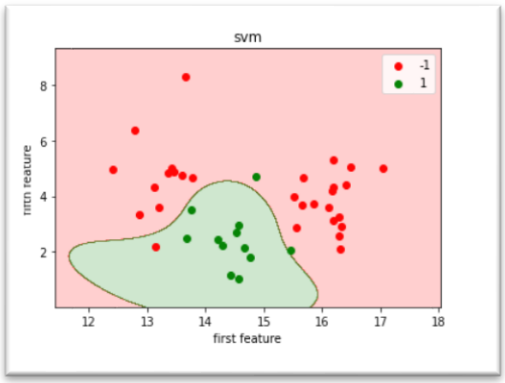



*Figure 10:Plotting SVM model with OVR*

- Applying perceptron with OVR (bainarized_labels) with each class in addition to obtaining confusion matrix, accuracy and plotting the decision boundary.

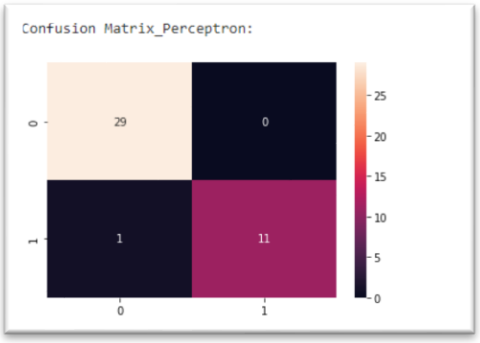



*Figure 11:confusion matrix on Perceptron model with OVR*

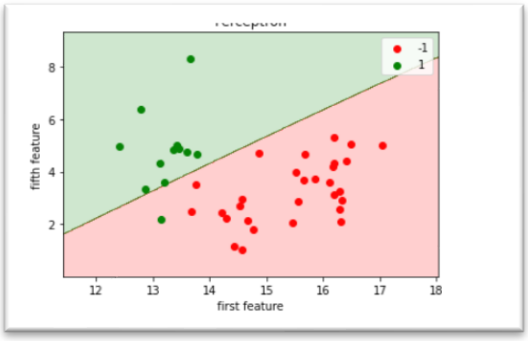



*Figure 12:confusion matrix on Perceptron model with OVR*

- After Applying the six models the best performance to SVM with class3 and perceptron with class2 and class3.
- Applying argmax function with SVM each class (OVR) to aggregate confidence scores and obtain the accuracy from three model of SVM, confusion matrix and plotting the correct and wrong prediction points.
- Using function plot_point to plot the correct and wrong prediction points.

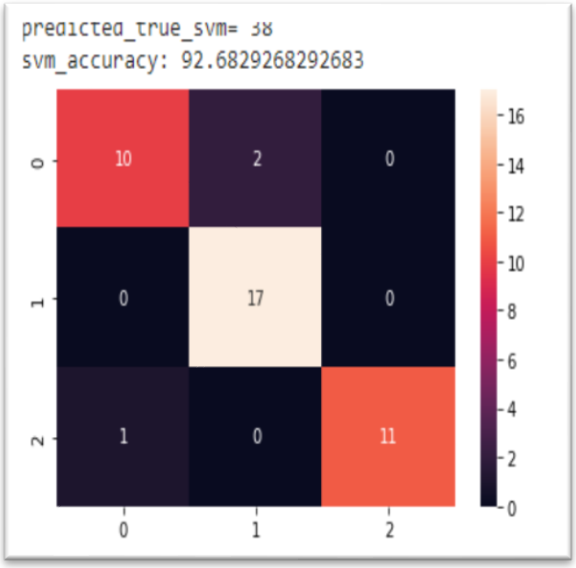



*Figure 13:confusion matrix argmax with svm model*

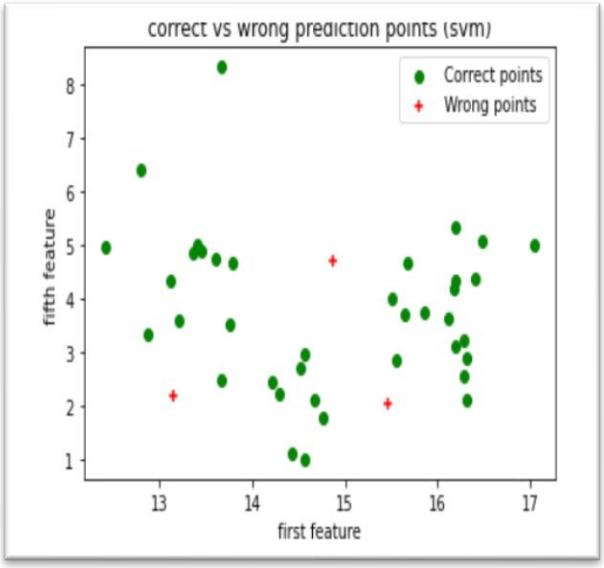



*Figure 14:plotting argmax with svm model*

- Applying argmax function with perceptron per each class (OVR) to aggregate confidence scores and obtain the accuracy from three model of perceptron, confusion matrix and plotting the correct and wrong prediction points.

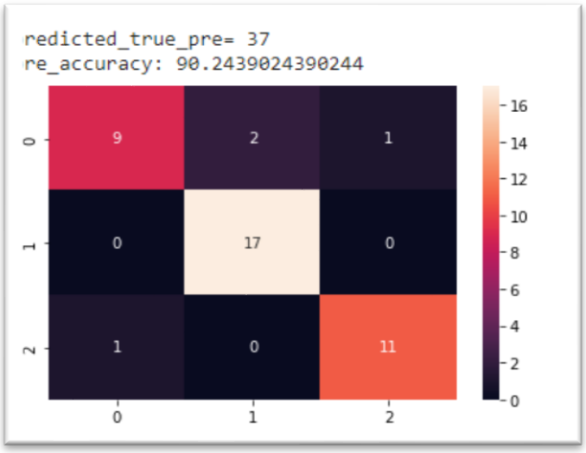



*Figure 15:confusion matrix argmax with perceptron model*

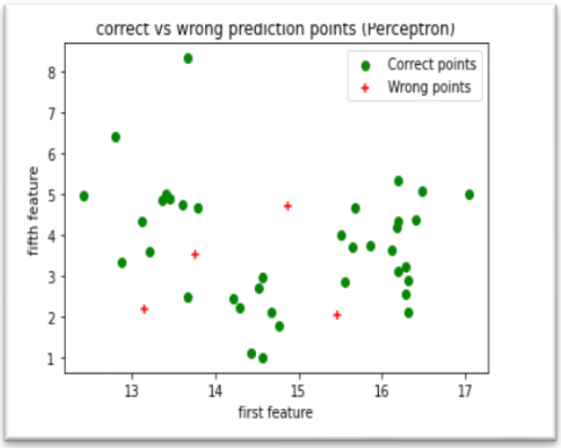



*Figure 16: confusion matrix argmax with perceptron model*

- Using new aggregation strategy after searching by stacking classifier. Stacking classifier uses the result from models to input the final model to obtain the performance of the final model. It predicts accuracy 92.68% with SVC and 90.24% with perceptron.

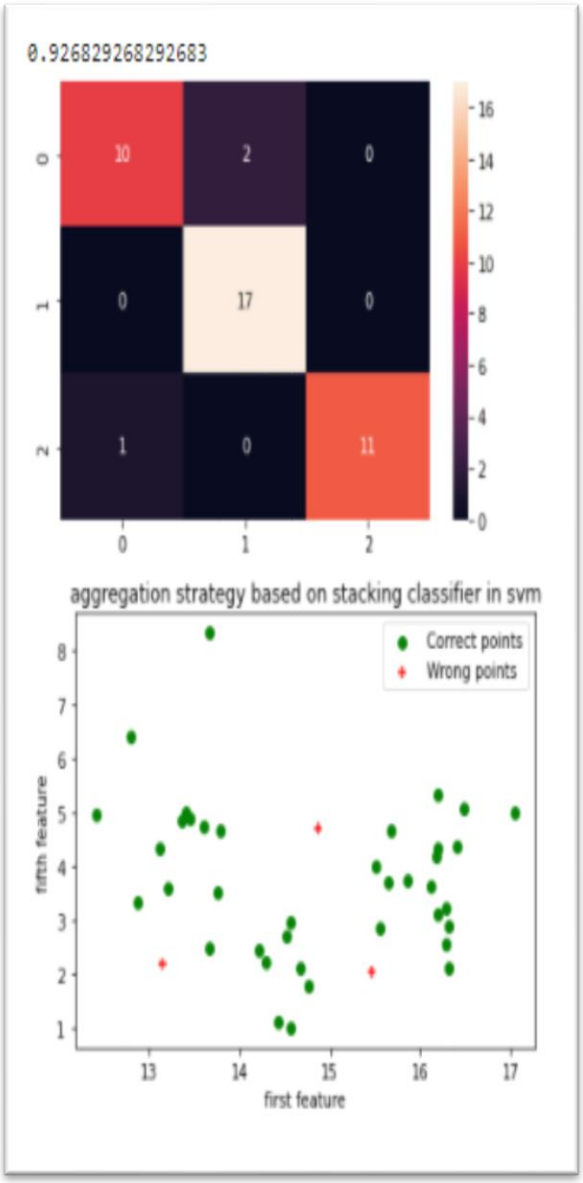



*Figure 17: Stacking classifier with svm model*

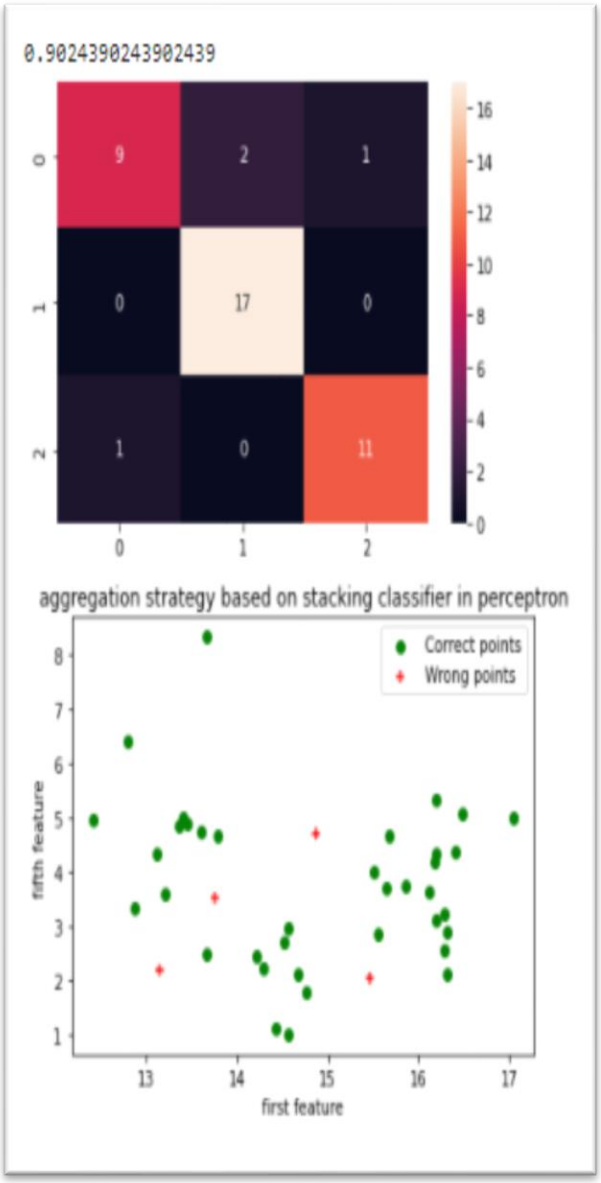



*Figure 18: Stacking classifier with perceptron model*

- Using another aggregation strategy like a voting classifier that train on an ensemble model and predicts an output based on their highest probability of chosen class as the output. It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. It predicts accuracy 92.68% with SVC and 95.12% with perceptron.

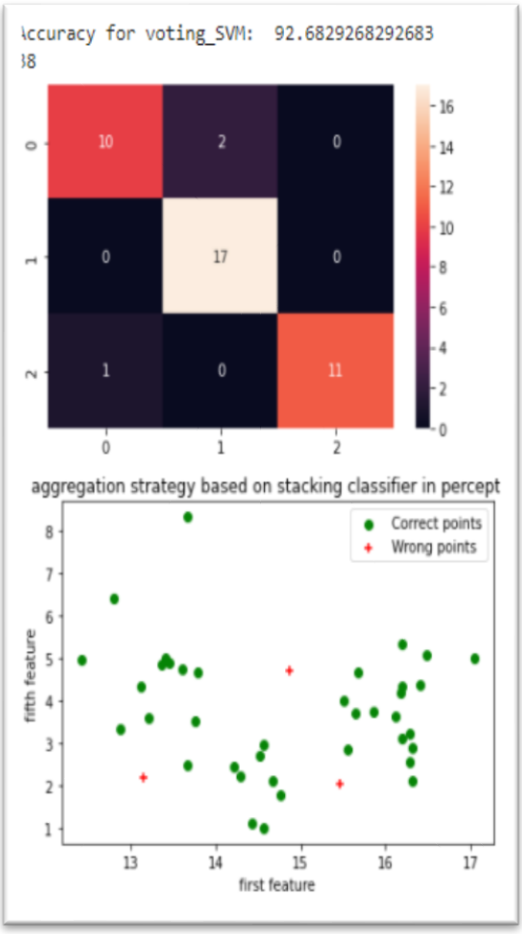



*Figure 19:voting classifier with svm model*

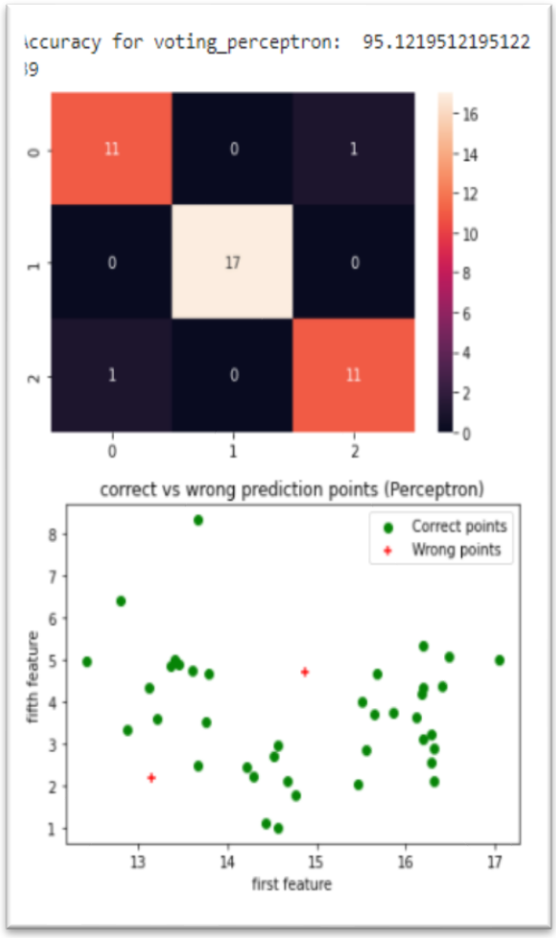



*Figure 20:classifier with perceptron model*

## 3. Conclusion

In conclusion, this report could be summarized by loading the data from seeds dataset, after that we split the data and applied SVM and Perceptron Algorithms on it. Then the data was binarized for applying OVR (1 for positive class, -1 for negative class), then the cleaned data applied in evaluating the models (argmax, stacking classifier and Voting classifier). Finally, it was found that using the Voting classifier with Perceptron model and SVC will give better performance of accuracy in prediction (92.68% with SVC and 95.12% with perceptron).