COMP3009/ COMP4139 Machine Learning

Lab 4 – Machine Learning Models (2)

Dr Xin Chen, Autumn Semester, 2024

1. Introduction

This lab session continues the work of lab 3 by implementing more nonlinear models. We will use the WDBC data, so it is important you complete lab 2 and 3 first. You may also use this lab session to work on Assignment 1 in your group.

2. Tutorial

2.1 Predictive model using Decision Trees

- **Use Decision Tree classifier to train a model. What does the parameter max-depth mean?**

```
from sklearn.tree import DecisionTreeClassifier
 from sklearn import tree

tree_clf = DecisionTreeClassifier(max_depth=2)
tree_clf.fit(Xs_train, y_train)
```

- **Predict the outcome of the test set. Why we have two columns in the output array?**

```
tree_clf.predict_proba(Xs_test)
```

- **Classify the test dataset and output the accuracy. Compare the accuracy to the previously trained linear models.**

```
classifier_score = tree_clf.score(Xs_test, y_test)
print('The classifier accuracy score of Decision Tree is
{:03.2f}'.format(classifier_score))
```

- **We can visualize the trained Decision Tree. Check the values of gini, samples, value, etc. Do you understand their meanings? Learn what "gini impurity" is. Observe if change the "max_depth" of the tree.**

```
tree.plot_tree(tree_clf)
```

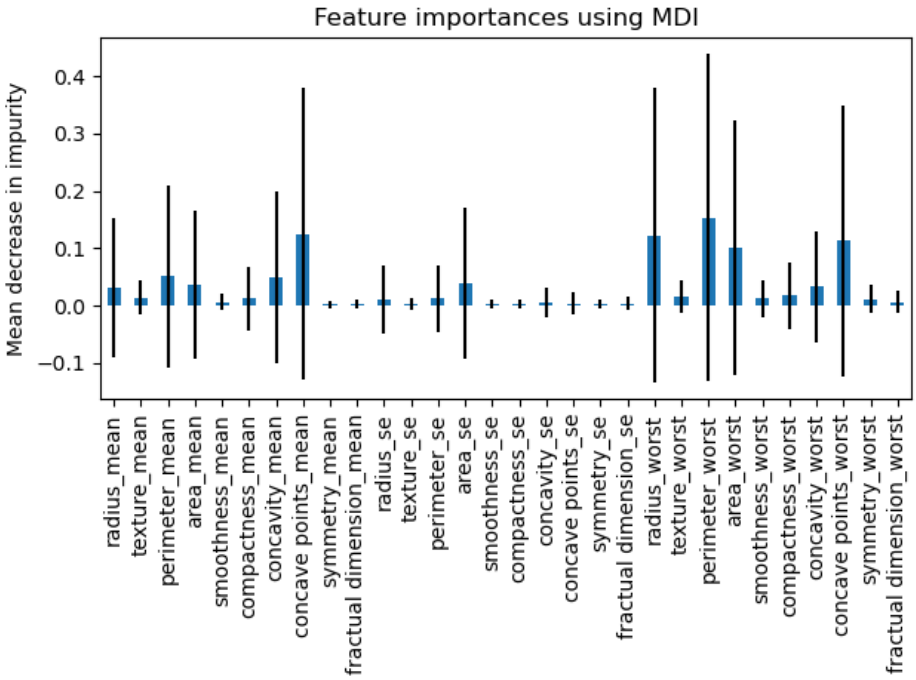
2.2 Predictive model using Random Forests

- **Use Random Forests classifier to perform the training and testing. Understand and play with the model parameters.**

```
from sklearn.ensemble import RandomForestClassifier

rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=10,
n_jobs=-1)
rnd_clf.fit(Xs_train, y_train)

y_pred_rf = rnd_clf.predict(Xs_test)

classifier_score = rnd_clf.score(Xs_test, y_test)
print('The classifier accuracy score of Random Forest is
{:03.2f}'.format(classifier_score))
```

- **Feature importance calculated by Random Forests. There are more than one method to rank the features. Check:**
  https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html

```
importances = rnd_clf.feature_importances_
std = np.std([tree.feature_importances_ for tree in rnd_clf.estimators_],
axis=0)
feature_names=all_df.columns[1:]

forest_importances = pd.Series(importances, index=feature_names)

fig, ax = plt.subplots()
forest_importances.plot.bar(yerr=std, ax=ax)
ax.set_title("Feature importance using MDI")
ax.set_ylabel("Mean decrease in impurity")
fig.tight_layout()
```



2.3 Predictive model using Multi-Layer Perceptron Neural Network.

- **Use MLP to achieve classification. Check the meaning of parameters of MLP:** **https://scikit- learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html**

```
from sklearn.neural_network import MLPClassifier

mlp_clf = MLPClassifier(random_state=1, max_iter=300).fit(Xs_train, y_train)
mlp_clf.predict_proba(Xs_test)

classifier_score = mlp_clf.score(Xs_test, y_test)
print('The classifier accuracy score of MLP is {:03.2f}'.format(classifier_score))
```

3. Additional Questions

- In the above implementations of Decision Tree and MLP, we used normalised feature values. Will we obtain similar performance by using the original feature values? Change the code to validate your answer.

- How to avoid overfitting using Decision Tree and ANN?