MGT 665 Lab 2

Long Lin

April 7, 2025

1 Abstract

In this lab, I use the Iris dataset to perform Logistic Regression, k-Nearest Neighbors, and Decision Trees. I then evaluate the performance of each model using accuracy, precision, recall, and F1-score to determine which model is the most effective. I was unable to determine which model performed the best, as all three classified the iris species perfectly.

2 Introduction

The objective of the study is to compare three popular types of classification models and their performance. I can use the results of this lab to determine which type of classification model is most effective for this type of data.

3 Related Work

I reference Chapter 3 of Dr. Itauma's book, "Machine Learning using Python", which discusses these classification models, to perform my own code. I also reference the LinkedIn Learning courses "Machine Learning with Python: Logistic Regression" and "Machine Learning with Python: Decision Trees" for additional coding support.

4 Methodology

4.1 Loading and Processing Data

[1]:
```
import pandas as pd
```

[2]:
```
iris = pd.read_csv("Iris.csv")
```

[3]:
```
iris.head()
```

[3]:

	Id	sepal length	sepal width	petal length	petal width	class
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa

[4]:
```
iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Id            150 non-null    int64
 1   sepal length  150 non-null    float64
 2   sepal width   150 non-null    float64
 3   petal length  150 non-null    float64
 4   petal width   150 non-null    float64
 5   class         150 non-null    object
dtypes: float64(4), int64(1), object(1)
memory usage: 7.2+ KB
```

The dataset does not have missing variables, and there is nothing to preprocess, so we will now visualize the data.

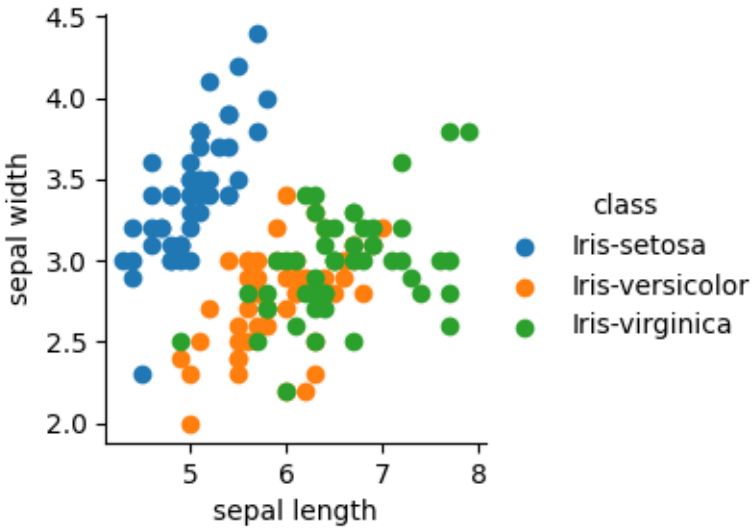
4.2 Exploratory Data Analysis

[5]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

Here we plot sepal length vs sepal width, separated by class.

[6]:
```
sns.FacetGrid(iris, hue="class") \
   .map(plt.scatter, "sepal length", "sepal width") \
   .add_legend()
```

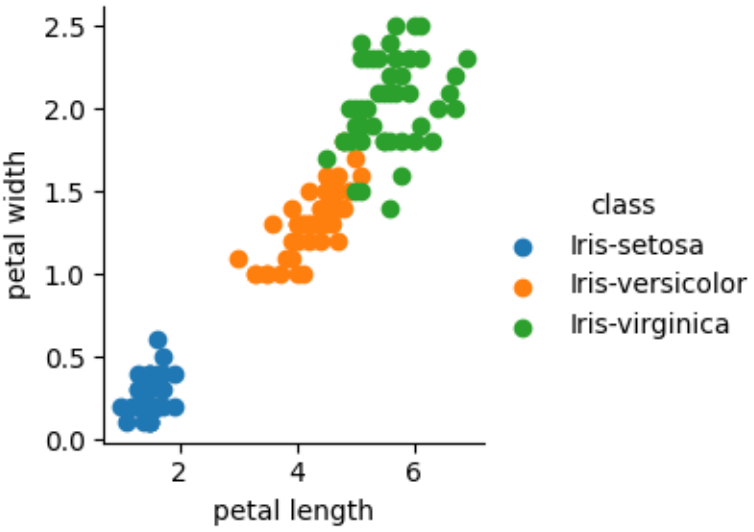
[6]: <seaborn.axisgrid.FacetGrid at 0x75d1b4ffff80>



We now plot petal length vs petal width, separated by class.

[7]:
```
sns.FacetGrid(iris, hue="class") \
   .map(plt.scatter, "petal length", "petal width") \
   .add_legend()
```

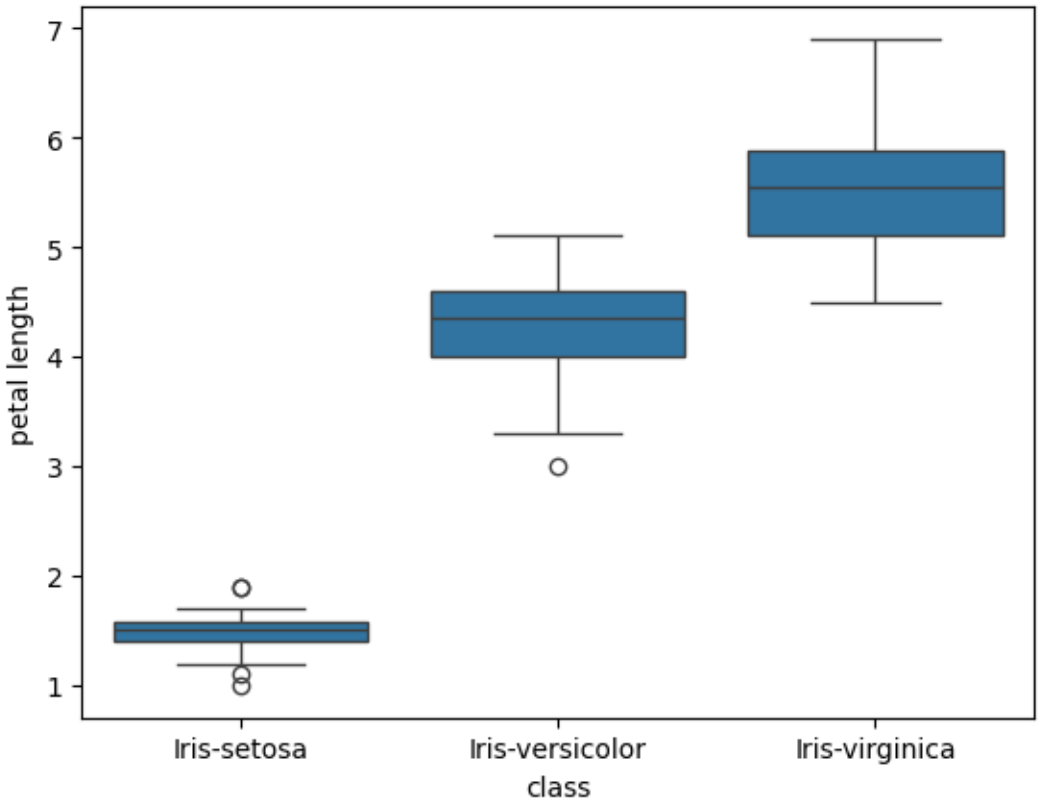
[7]:
```
<seaborn.axisgrid.FacetGrid at 0x75d1b4a07a70>
```



Now, we look at boxplots for each variable by class.

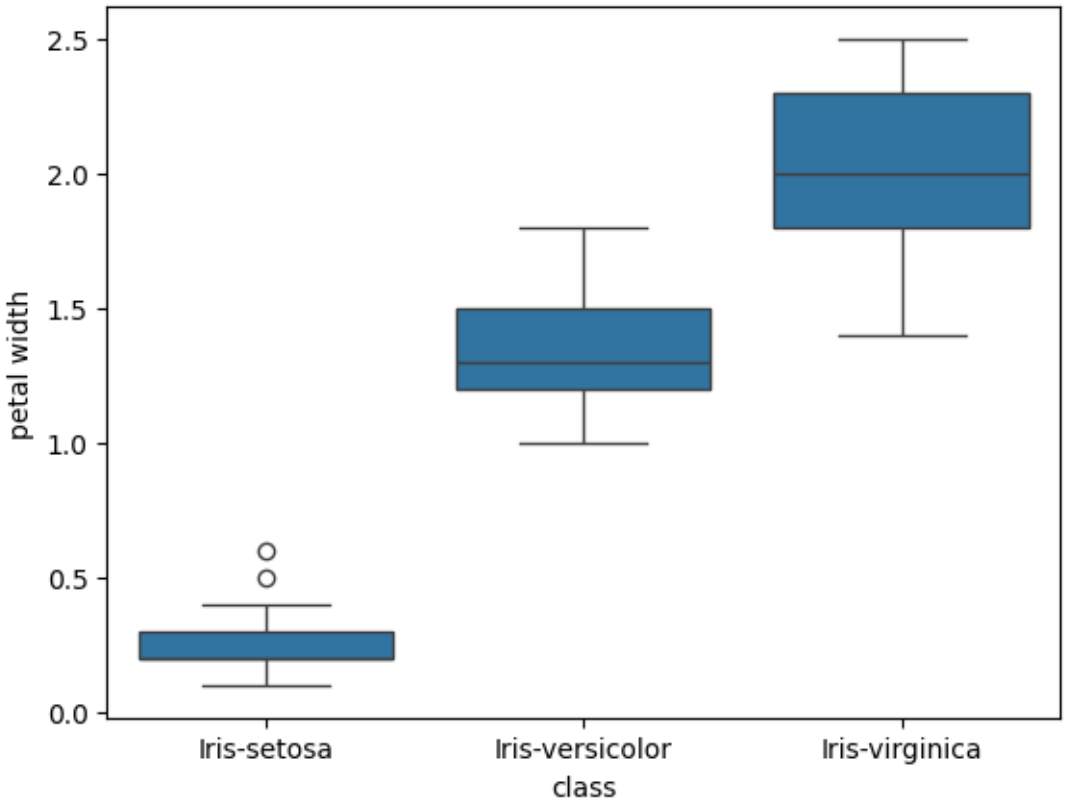
[8]:
```
sns.boxplot(x="class", y="petal length", data=iris) ## Petal Length
```

[8]:
```
<Axes: xlabel='class', ylabel='petal length'>
```



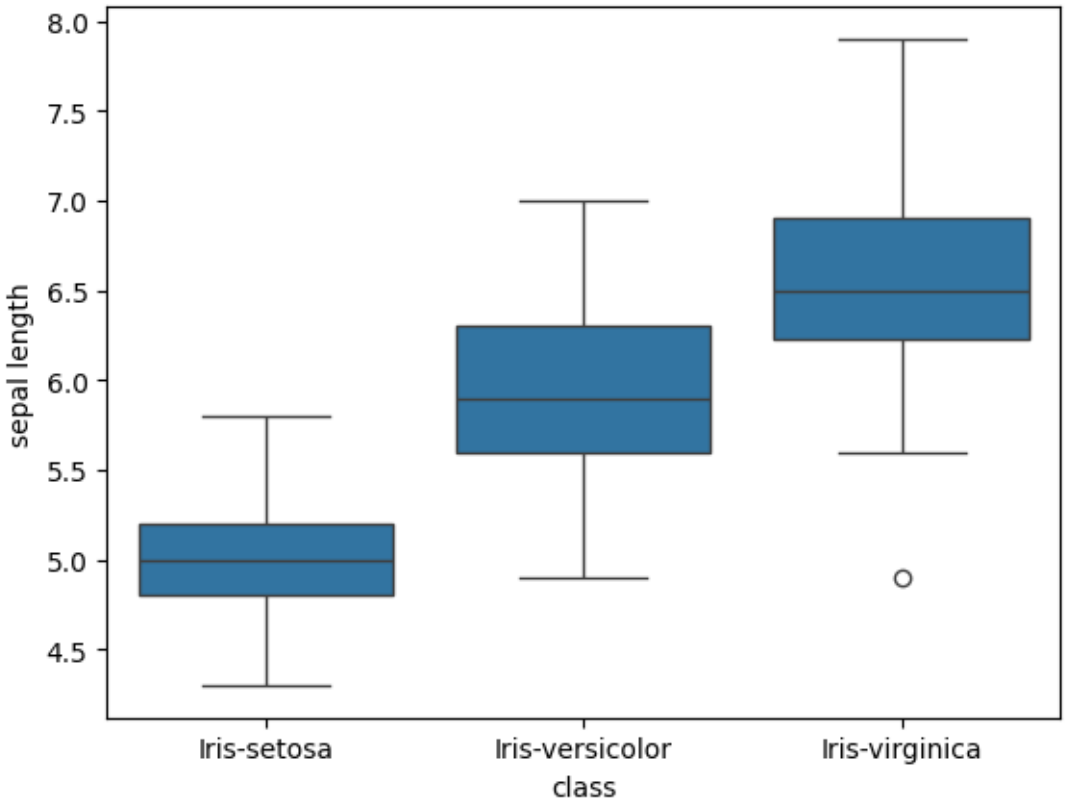
[9]:
```
sns.boxplot(x="class", y="petal width", data=iris) ## Petal Width
```

[9]:
```
<Axes: xlabel='class', ylabel='petal width'>
```



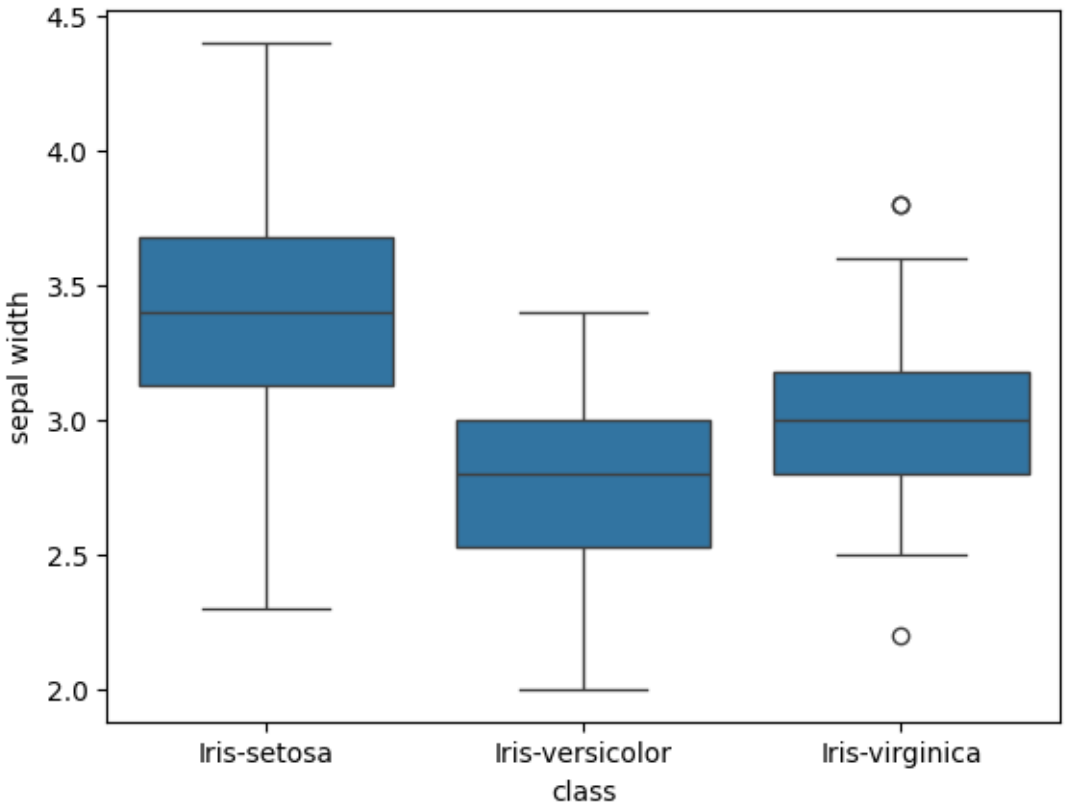
[10]:
```
sns.boxplot(x="class", y="sepal length", data=iris) ## Sepal Length
```

[10]:
```
<Axes: xlabel='class', ylabel='sepal length'>
```



[11]:
```
sns.boxplot(x="class", y="sepal width", data=iris) ## Sepal Width
```

[11]:
```
<Axes: xlabel='class', ylabel='sepal width'>
```



Overall, it seems like the relationship between petal length and width are useful in classifying the iris species, as well as petal length and width themselves, and the sepal length.

4.3 Classification Models

First, we will split the dataset into x and y, and then into training and testing sets.

[12]:
```
X = iris.loc[:, ['sepal length','sepal width', 'petal length', 'petal width',]].
  ↪values
y = iris.loc[:, 'class'].values
```

[13]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
  ↪random_state=42)
```

4.3.1 Logistic Regression

[14]:
```
from sklearn.linear_model import LogisticRegression

logistic = LogisticRegression(random_state=42)
logistic.fit(X_train, y_train)
```

[14]:
```
LogisticRegression(random_state=42)
```

[15]:
```
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score

ypred_logistic = logistic.predict(X_test)
```

[16]:
```
print("The accuracy is", accuracy_score(y_test, ypred_logistic))
print(classification_report(y_test, ypred_logistic))
```

```
The accuracy is 1.0
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

4.3.2 k-Nearest Neighbors

[17]:
```
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(X_train, y_train)
```

[17]:
```
KNeighborsClassifier(n_neighbors=3)
```

[18]:
```
ypred_knn = knn.predict(X_test)
```

[19]:
```
print("The accuracy is", accuracy_score(y_test, ypred_knn))
print(classification_report(y_test, ypred_knn))
```

```
The accuracy is 1.0
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

4.3.3 Decision Tree

[20]:
```
from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier(random_state = 42)
tree.fit(X_train, y_train)
```

[20]:
```
DecisionTreeClassifier(random_state=42)
```

[21]:
```
ypred_tree = tree.predict(X_test)
```

[22]:
```
print("The accuracy is", accuracy_score(y_test, ypred_tree))
print(classification_report(y_test, ypred_tree))
```

```
The accuracy is 1.0
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        10
Iris-versicolor       1.00      1.00      1.00         9
 Iris-virginica       1.00      1.00      1.00        11

       accuracy                           1.00        30
      macro avg       1.00      1.00      1.00        30
   weighted avg       1.00      1.00      1.00        30
```

5 Results

All three of the classification models used in this lab resulted in perfect accuracy, precision, recall, and F1-score. Thus, it is impossible to compare on which one was the most effective in predicting the iris species in the dataset.

6 Discussion

The results show that all three classification models were perfectly effective in predicting the iris species in the dataset. This shows that all three models can be used to great success when predicting a categorical variable using numeric variables, like in this case. However, in order to properly determine which model was the most effective, perhaps a different dataset should be used, such as one with more variables, or one with messier data. This can perhaps better show which model is best at dealing with different things in the data.

7 Conclusion

Overall, Logistic Regression, k-Nearest Neighbors, and Decision Tree were all effective in classifying the different iris species using sepal length, sepal width, petal length, and petal width. Further research with different datasets can be conducted to determine which model is most effective for which type of dataset.

8 References

Itauma, I. (n.d.). 3 Chapter 3: Supervised learning - classification. Machine Learning using Python - 3 Chapter 3: Supervised Learning - Classification. https://amightyo.quarto.pub/machine-learning-using-python/Chapter_3.html

Nwanganga, F. (2022, May 20). How to build a classification tree in Python - python video tutorial: Linkedin learning, formerly Lynda.com. LinkedIn. https://www.linkedin.com/learning/machine-learning-with-python-decision-trees/how-to-build-a-classification-tree-in-python?u=279222306

Nwanganga, F. (2022b, November 9). Classifying data with logistic regression - python video tutorial: Linkedin learning, formerly Lynda.com. LinkedIn. https://www.linkedin.com/learning/machine-learning-with-python-logistic-regression/classifying-data-with-logistic-regression?u=279222306