Data Analyst Internship Report

Introduction

Name	Vedant Maladkar
Project Title	Iris classification
Domain	Data Analyst
Tools	Python, Machine Learning

Iris classification

Objective-

The aim of this project involves creating a Logistic Regression model to classify iris flowers into three species (Setosa, Versicolour, and Virginica) based on the length and width of their petals and sepals.

Problem Statement-

- The model should achieve a high level of accuracy in classifying iris species.
- The model's predictions should be consistent and reliable, as measured by cross-validation.

Dataset Description-

This dataset is named ‘Iris’. It contains a total of 150 entries. The columns of this dataset are 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm', 'Species'

Exploratory Data Analysis (EDA)-

```
df.info()
✓ 0.1s
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   SepalLengthCm  150 non-null    float64
 1   SepalWidthCm   150 non-null    float64
 2   PetalLengthCm  150 non-null    float64
 3   PetalWidthCm   150 non-null    float64
 4   Species        150 non-null    object
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

Seeing this we can say that the there are no null values and the dataset is already cleaned.

Feature Engineering-

```
label_encoder = LabelEncoder()
df['Species'] = label_encoder.fit_transform(df['Species'])
```

✓ 0.0s

Using this the code automatically converts the 'Species' string values to the numerical values (iris-setosa, iris-virsicolour, iris-viginica) as 0, 1, 2

```
scalar=StandardScaler()
df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]=scalar.fit_transform(df[['SepalLengthCm',
 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']])
```

✓ 0.0s Python

This normalized 'SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm' using Standard Scalar.

```
X = df[['SepalLengthCm', 'SepalWidthCm', 'PetalLengthCm', 'PetalWidthCm']]
X = sm.add_constant(x)

vif_data = pd.DataFrame()
vif_data['feature'] = X.columns
vif_data['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

print(vif_data)
```

✓ 0.0s

```
        feature        VIF
0         const   1.000000
1  SepalLengthCm   7.103113
2   SepalWidthCm   2.099039
3  PetalLengthCm  31.397292
4   PetalWidthCm  16.141564
```

Here we are removing 'PetalLengthCm' since the value is multicollinear. After removing the column we get

```
X = df[['SepalLengthCm', 'SepalWidthCm', 'PetalWidthCm']]
X = sm.add_constant(X)

vif_data = pd.DataFrame()
vif_data['feature'] = X.columns
vif_data['VIF'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
print(vif_data)
```

148] ✓ 0.0s

```
         feature       VIF
0          const  1.000000
1  SepalLengthCm  3.414225
2   SepalWidthCm  1.294507
3   PetalWidthCm  3.864678
```

Model Building-

Here we use Logistic Regression and divide dataset to 80-20 for training and testing sets.

```
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

✓ 0.0s

Generate + Code + Markdown

```
model = LogisticRegression()
model.fit(x_train, y_train)
y_pred = model.predict(x_test)
```

✓ 0.0s

Model evaluation includes:

- Accuracy Score
- Confusion matrix
- R2-Score
- Mean Squared Error
- Classification report
- K-Fold Cross Validation

Results-

```
accuracy_score(y_test, y_pred)
```
✓ 0.0s
```
0.9666666666666667
```

```
cm=confusion_matrix(y_test, y_pred)
cm
```
✓ 0.0s
```
array([[10,  0,  0],
       [ 0,  8,  1],
       [ 0,  0, 11]])
```
Generate

```
r2_score(y_test,y_pred)
```
✓ 0.0s
```
0.9523052464228935
```

```
mse = mean_squared_error(y_test, y_pred)
print(mse)
```
✓ 0.0s
```
0.03333333333333333
```

```
cr=classification_report(y_test,y_pred, target_names=['Setosa', 'versicolor', 'virginica'])
print(cr)
```
✓ 0.0s
```
              precision    recall  f1-score   support

      Setosa       1.00      1.00      1.00        10
  versicolor       1.00      0.89      0.94         9
   virginica       0.92      1.00      0.96        11

    accuracy                           0.97        30
   macro avg       0.97      0.96      0.97        30
weighted avg       0.97      0.97      0.97        30
```

```
model = LogisticRegression()
K=5
kfold = KFold(K,random_state=0,shuffle=True)
mse_cv = cross_val_score(model,x,y,cv=kfold,scoring='accuracy')
print(mse_cv)
```

✓ 0.1s

```
[1.         0.9        1.         1.         0.93333333]
```

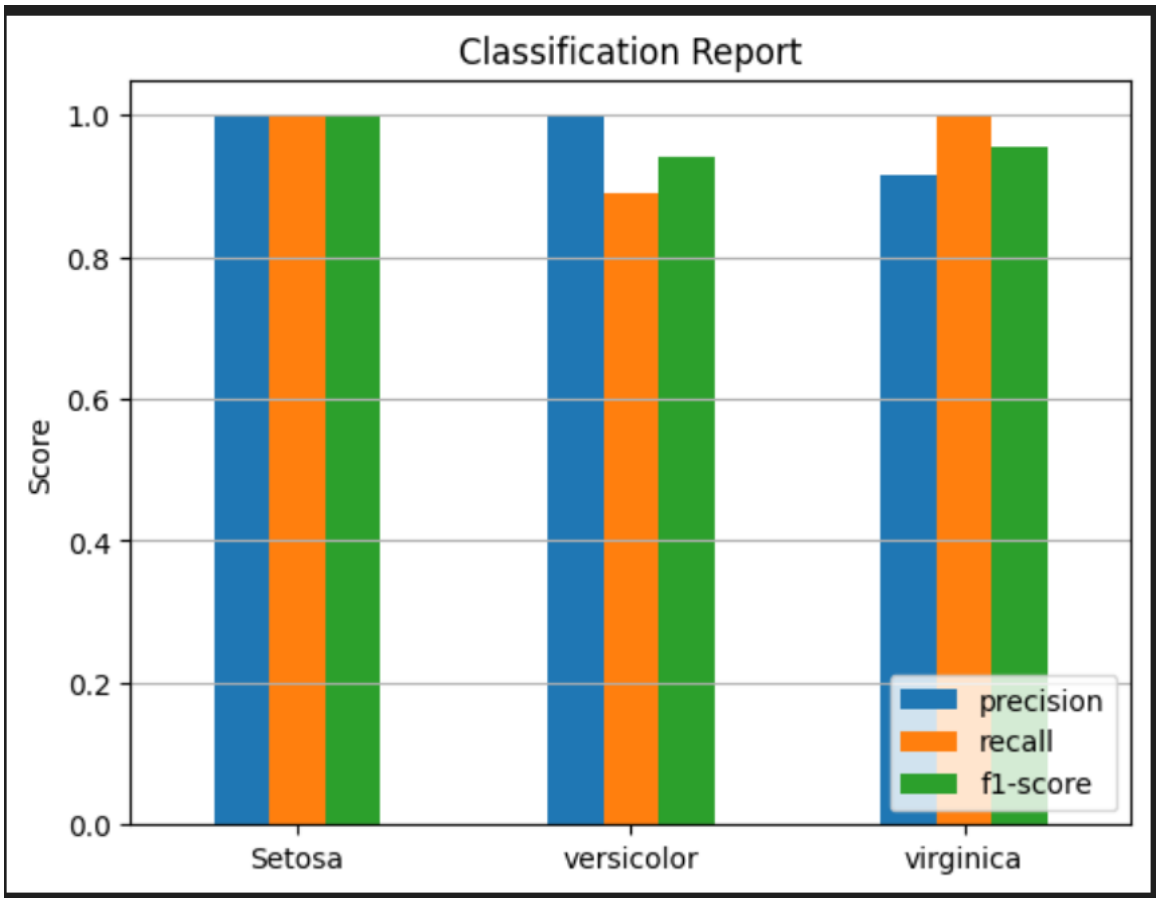
Logistic model accuracy is approximately 97%

Model evaluation shows:

- R2 score is approximately 0.95 showing strong model fit
- Mean Squared Error is 0.033 indicating low prediction error
- Setosa and Virginica showed no false negatives or false positives which means it predicted accurately with perfect recall
- Versicolor showed one error
- The 5 fold cross-validation conforms that I have an average accuracy of 96.67% conforming that the Logistic Regression model performs consistently across different subsets of iris.

Plot-

This plot show the classification report setosa, versicolor and virginica.



Summary-

- The logistic regression model classifies the iris species with high accuracy and perception of 96.67% showing strong performance.
- This report showed that setosa and virginica have perfect classification and versicolor have a very high classification.
- This is further verified using the 5-fold cross-validation where average accuracy is 96.67%.
- This project makes us understand how reliable is logistic regression for multi-class classification and has a strong foundation for more advanced machine learning applications.

Reference-

- Libraries - pandas, matplotlib, scikit-learn, statsmodels.
- Tools – Visual Studio Code