



Principles of Data Mining and Machine Learning

Log Book

Module Code: MOD007892

Academic Year: 2023/2024

SID: 2224089

Contents

Week 1	3
Week 2	4
Q1.....	4
Q2.....	4
Week 3	5
Q1.....	5
Q2.....	5
Week 4	6
Q1.....	6
Q2.....	6
Week 5	7
Q1.....	7
Week 6	8
Q1.....	8
Q2.....	9
Week 7	10
Q1.....	10
Q2.....	10
Week 8	11
Q1.....	11
Week 9	12
Q1.....	12
Q2.....	12
Week 10	13
Q1.....	13
Q2.....	13
Q3.....	13
Q4.....	13
Q5.....	13

GITHUB Link Can found Below

https://github.com/chandima131/DM_LabTutorials_2224089.git

Week 1

Data Frame:

The Pandas DataFrame is a 2D tabular data structure with labeled columns and rows that can be resized and contain heterogeneous data. It is highly efficient in performing data manipulation and analysis, making it the ideal choice for EDA tasks.

Group By:

The Pandas library has a class called GroupBy, which is used for grouping data based on certain criteria. It allows to apply a function to each group separately, which is very helpful when you want to analyze and summarize data based on specific features.

Categorical:

The Categorical class is used to represent categorical data, which provides a more efficient way to store and analyze discrete data with a fixed and known set of categories. It is particularly useful in EDA for working with categorical variables in telecom data, such as customer types or service plans.

Summary Tables:

Generating summary tables helps aggregate and present key statistics through pivot tables. They offer a quick overview of metrics like total revenue and average call duration in the dataset.

Apply:

The apply() function enables custom functions to be applied to each element or group in a DataFrame, allowing for complex transformations or computations on telecom data and improving interpretability.

Week 2

Q1.

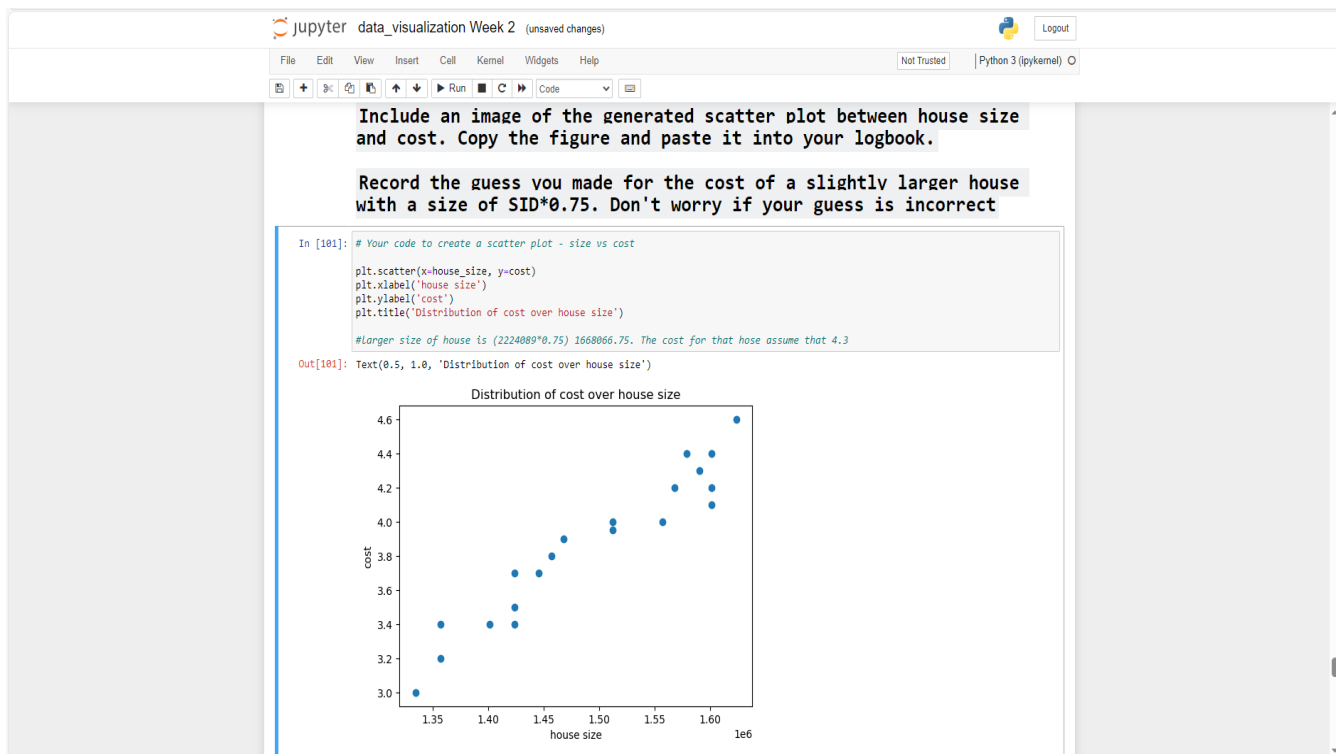


Figure 2.1

Q2.

SID: 2224089

$$\text{SID} \times 0.75 \Rightarrow 2224089 \times 0.75 = 1668066.75$$

Since according to the scatter diagram, the cost for the size of 1668066.75 assume that 4.3.

Week 3

Q1.

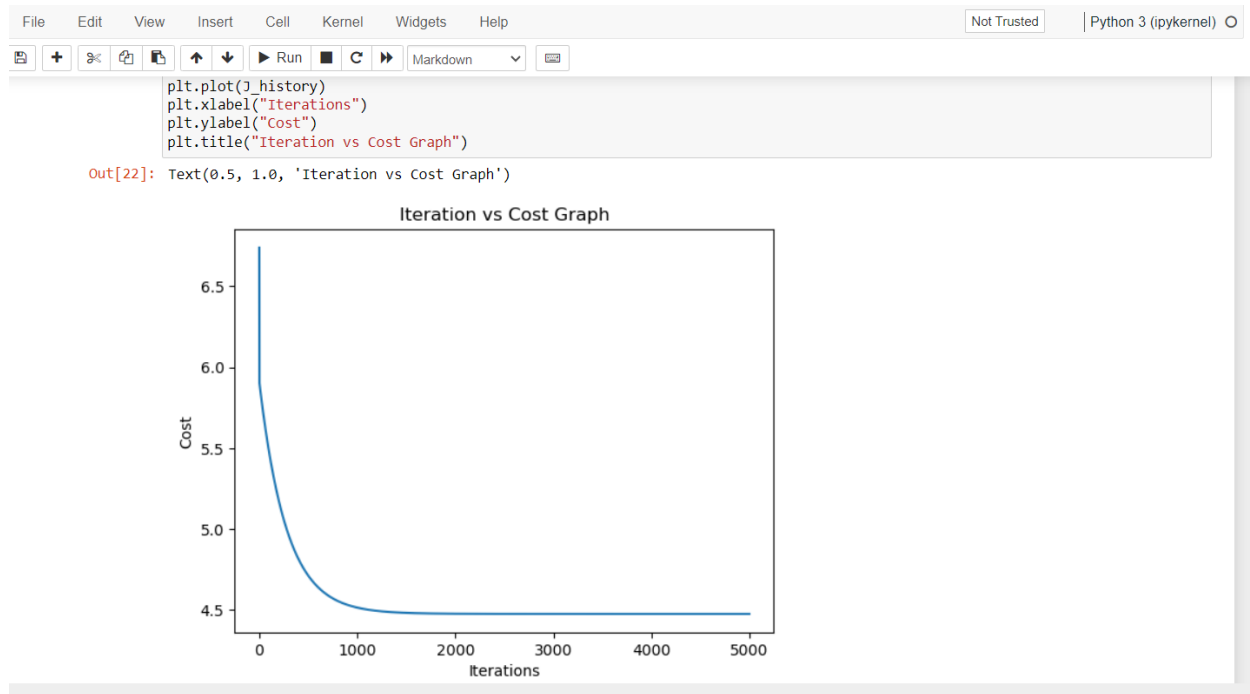


Figure 3.1 Answer for Question 1

Q2

Important: You need to provide the output values of the next cell i.e. predictions for unknown values in your lablogbook.

```
In [18]: SID = 2224089
First_City = SID/10 # Put the population of first city as 10 times less than your SID
Second_City = SID/30 # Put the population of second city as 30 times less than your SID

predict1 = (prediction([1, First_City/10000]),(new_theta))
predict2 = (prediction([1, Second_City/10000]),(new_theta))

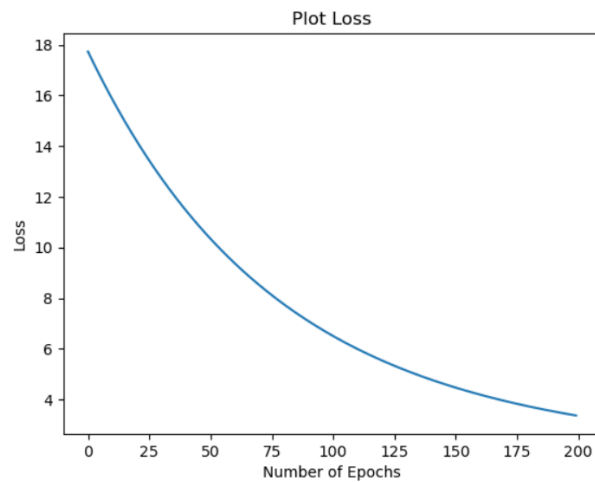
print(f'For a population of {First_City} people, profit will be {predict1[0]}')
print(f'For a population of {Second_City} people, profit will be {predict2[0]}')
```

For a population of 222408.9 people, profit will be 226377.5623204882
For a population of 74136.3 people, profit will be 49490.51736911497

Figure 3.2 Answer for Question 2

Week 4

Q1.



4.1 The loss function

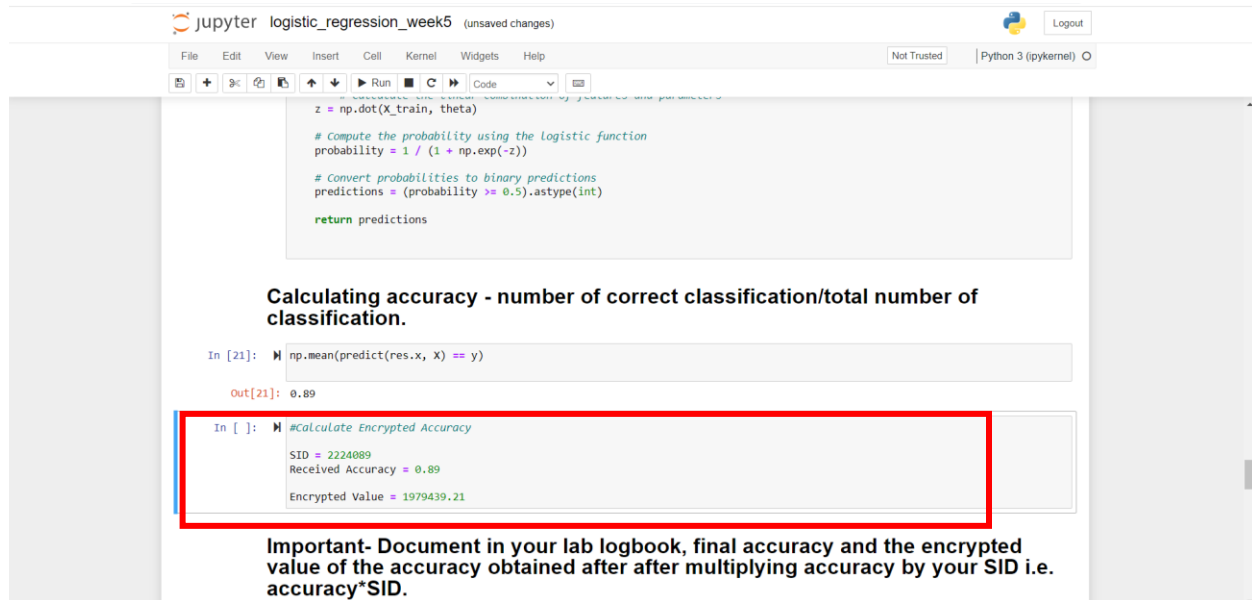
Q2.

Perform the predictions on X_tset. Call predict method.

```
In [42]: # Your code to predict the profit i.e. y values  
predictor = regressor.predict(w_trained,[350000,840000,8900000,1,0,1])  
print(predictor)  
3244823.4086816576
```

Week 5

Q1.



The image shows a Jupyter Notebook interface for a file named 'logistic_regression_week5'. The notebook contains a code cell with the following Python code:

```
z = np.dot(X_train, theta)

# Compute the probability using the Logistic function
probability = 1 / (1 + np.exp(-z))

# Convert probabilities to binary predictions
predictions = (probability > 0.5).astype(int)

return predictions
```

Below the code cell, the text "Calculating accuracy - number of correct classification/total number of classification." is displayed. The next code cell shows the calculation of accuracy:

```
In [21]: np.mean(predict(res.x, X) == y)
```

The output of this cell is 0.89. The following code cell, which is highlighted with a red border, calculates the encrypted accuracy:

```
In [ ]: #Calculate Encrypted Accuracy

SID = 2224009
Received Accuracy = 0.89
Encrypted Value = 1979439.21
```

Below this cell, the text "Important- Document in your lab logbook, final accuracy and the encrypted value of the accuracy obtained after after multiplying accuracy by your SID i.e. accuracy*SID." is displayed.

Figure 5.1 Answer for Encrypted Accuracy

Week 6

Q1.

```
In [19]: # Your code to compare the different models and bar plot comparing model accuracy.
# You can split your code in multiple cells
#
#clf = MLPClassifier(hidden_layer_sizes=(500, 500,500), max_iter=2000)
clf = MLPClassifier(solver='lbfgs',
                    alpha=1e-5,
                    hidden_layer_sizes=(6,),
                    random_state=1)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
report = classification_report(y_test, y_pred)
print(report)
```

	precision	recall	f1-score	support
0	0.75	0.97	0.84	89
1	0.56	0.27	0.36	90
2	0.00	0.00	0.00	92
3	0.18	0.97	0.31	93
4	0.04	0.96	0.95	76
5	0.00	0.00	0.00	108
6	0.99	0.85	0.92	89
7	0.93	0.91	0.92	78
8	0.29	0.04	0.08	92
9	0.00	0.00	0.00	92
accuracy			0.47	899
macro avg	0.46	0.50	0.44	899
weighted avg	0.44	0.47	0.41	899

C:\Users\chand\anaconda3\Lib\site-packages\sklearn\normal_network\multilayer_perceptron.py:546: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

6.1 MLP Classifier Accuracy by changing solver.

```
In [23]: #clf = MLPClassifier(solver='sgd',
#                    alpha=1e-5,
#                    hidden_layer_sizes=(6,),
#                    random_state=1,
#                    max_iter=2000)
clf = MLPClassifier(solver='sgd',
                    alpha=1e-5,
                    hidden_layer_sizes=(6,),
                    random_state=1,
                    max_iter=2000)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
report = classification_report(y_test, y_pred)
print(report)
```

	precision	recall	f1-score	support
0	0.90	0.93	0.92	89
1	0.82	0.80	0.81	90
2	0.89	0.91	0.90	92
3	0.87	0.86	0.86	93
4	0.87	0.95	0.91	76
5	0.85	0.77	0.81	108
6	0.94	0.93	0.94	89
7	0.81	0.97	0.88	78
8	0.88	0.73	0.80	92
9	0.73	0.75	0.74	92
accuracy			0.86	899
macro avg	0.86	0.86	0.86	899
weighted avg	0.86	0.86	0.85	899

C:\Users\chand\anaconda3\Lib\site-packages\sklearn\normal_network\multilayer_perceptron.py:691: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (2000) reached and the optimization hasn't converged yet.

6.2 MLP Classifier Accuracy by changing maximum iteration.


```
tic optimizer: maximum iterations (2000) reached and the optimization hasn't converged yet.
warnings.warn(
```

```
In [25]: clf = MLPClassifier(solver='adam',
                             alpha=1e-5,
                             hidden_layer_sizes=(1000,1000),
                             random_state=1,
                             max_iter=2000)
clf.fit(X_train, y_train)
y_pred = clf.predict(X_test)
report = classification_report(y_test, y_pred)
print(report)
```

	precision	recall	f1-score	support
0	0.99	1.00	0.99	89
1	0.96	0.99	0.97	90
2	1.00	0.99	0.99	92
3	0.98	0.99	0.98	93
4	0.99	1.00	0.99	76
5	0.95	0.96	0.95	108
6	0.98	0.98	0.98	89
7	1.00	1.00	1.00	78
8	0.99	0.91	0.95	92
9	0.98	0.98	0.98	92
accuracy			0.98	899
macro avg	0.98	0.98	0.98	899
weighted avg	0.98	0.98	0.98	899

6.3 MLP Classifier Accuracy changing solver and hidden layer size.

Q2.

```
In [30]: import sys
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.ticker import StrMethodFormatter
import numpy as np
plt.rcParams.defaults()

Fams = ['Answer1', 'Answer2', 'Answer3', 'Answer4']
AllTested = [0.97, 0.46, 0.86, 0.98]

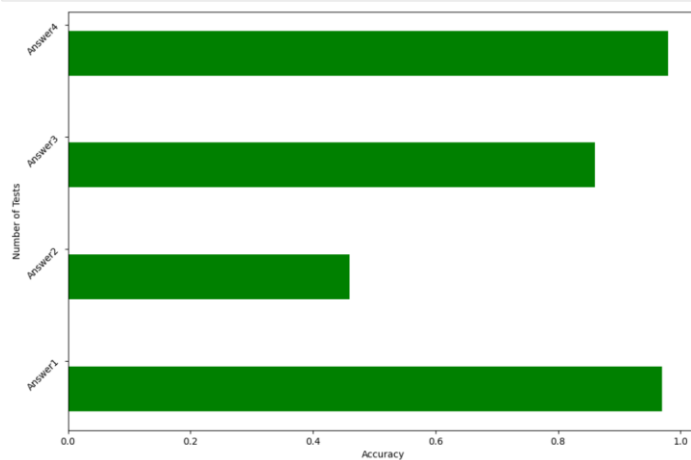
plt.figure(figsize=[12, 8])
x = np.arange(len(Fams))
plt.barh(x-0.25, AllTested, color='g', height=0.4)

plt.yticks([i for i in range(len(Fams))], Fams, rotation=45)

# Naming the x and y axis
plt.xlabel('Accuracy')
plt.ylabel('Number of Tests')

plt.savefig('day2.png', format='png')
```

6.2.1. Code for bar graph



6.2.2 Bar graph for Comparing answers.

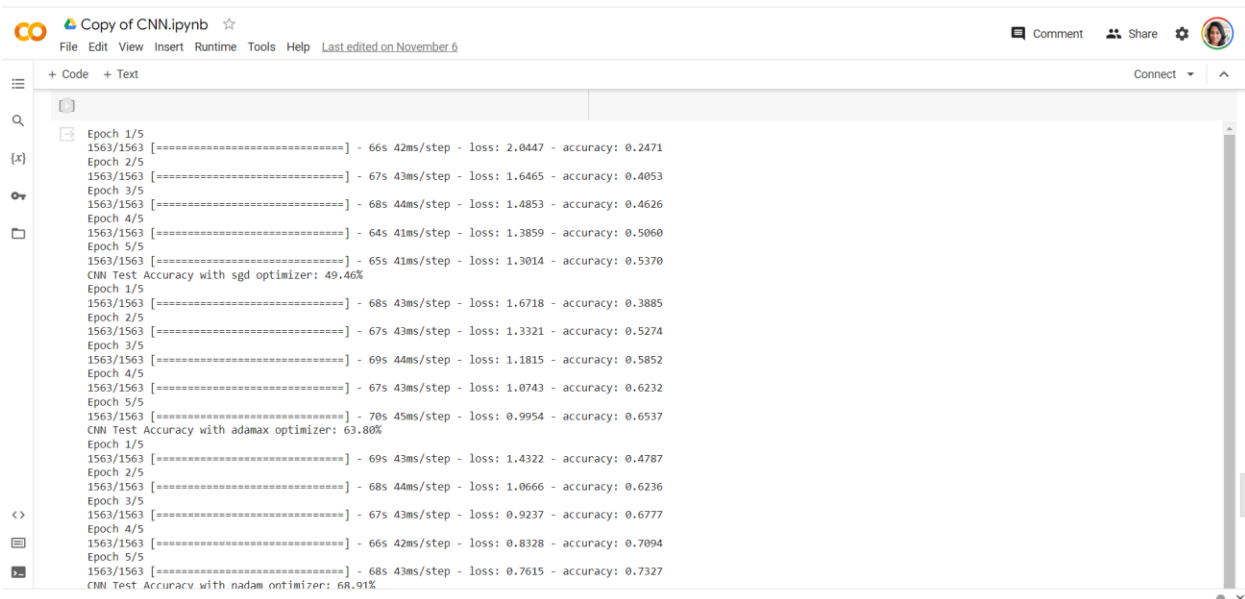
Week 7

Q1.



7.1 Bar graph depicting the comparison of training time.

Q2.



7.2 Final Accuracy Achieve

Week 8

Q1.

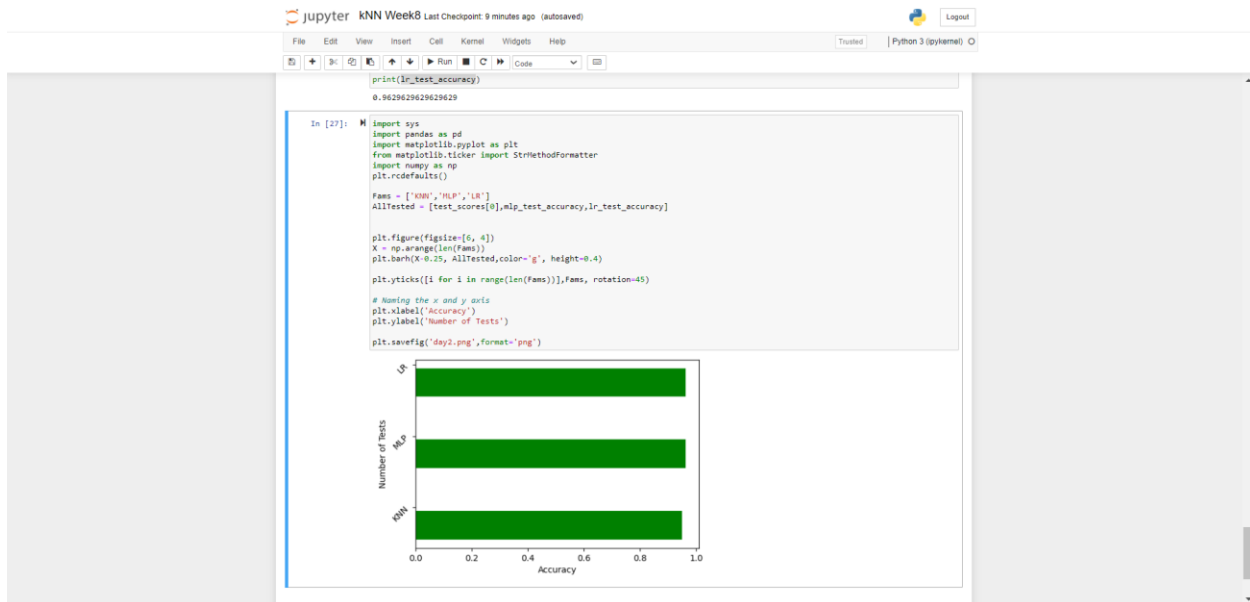
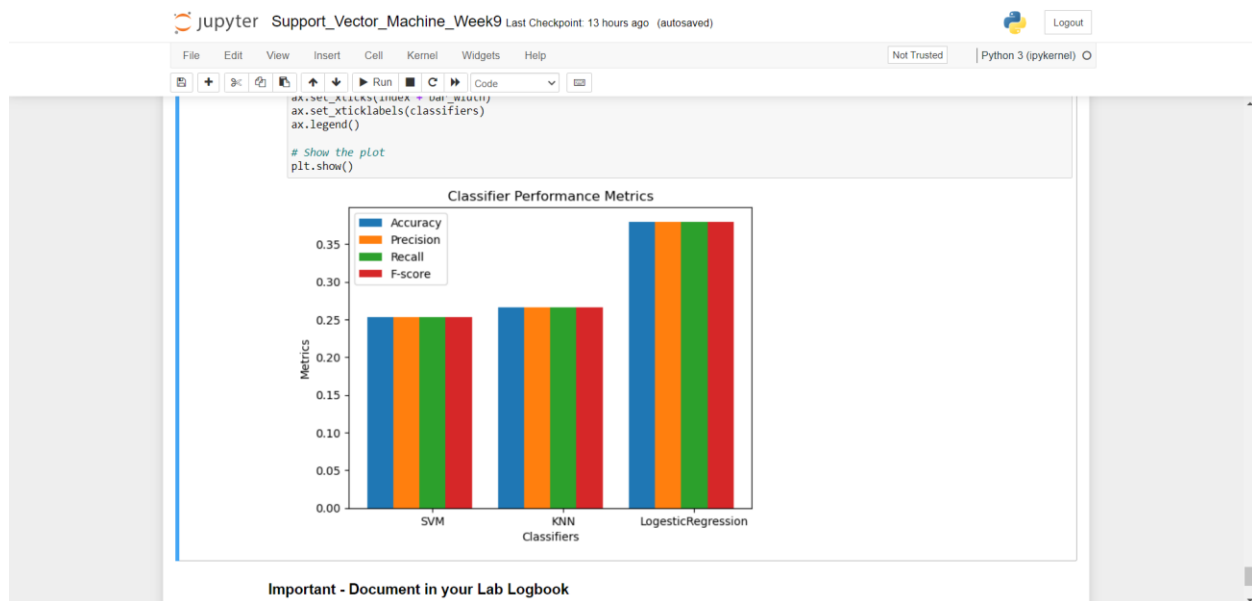


Figure8.1 Bar graph for comparing accuracy of KNN, MLP, LR

Week 9

Q1.



9.1 Comparing the performance of SVM, KNN, LR

Q2.

Best performance Algorithm is Logistic Regression. According to the graph LR shows higher performance in Accuracy, Precision, Recall, F-score.

Week 10

Q1.

The decision criterion at each node of the decision tree is based on a feature that optimally separates the data, with the goal of maximizing information gain or purity, depending on the chosen metric (entropy, Gini index, etc.).

Q2.

At each node, the algorithm selects the feature that reduces uncertainty or impurity, leading to more homogeneous child nodes as the entropy, Gini index, or loss changes.

Q3.

As we move down the tree, entropy decreases and subsets become better separated, reducing uncertainty between classes.

Q4.

The decision tree narrows down the subset of data that satisfies its conditions at each node, creating a more specific and refined pattern as it progresses.

Q5.

The leaf nodes represent the final prediction based on the characteristics along the path from the root.