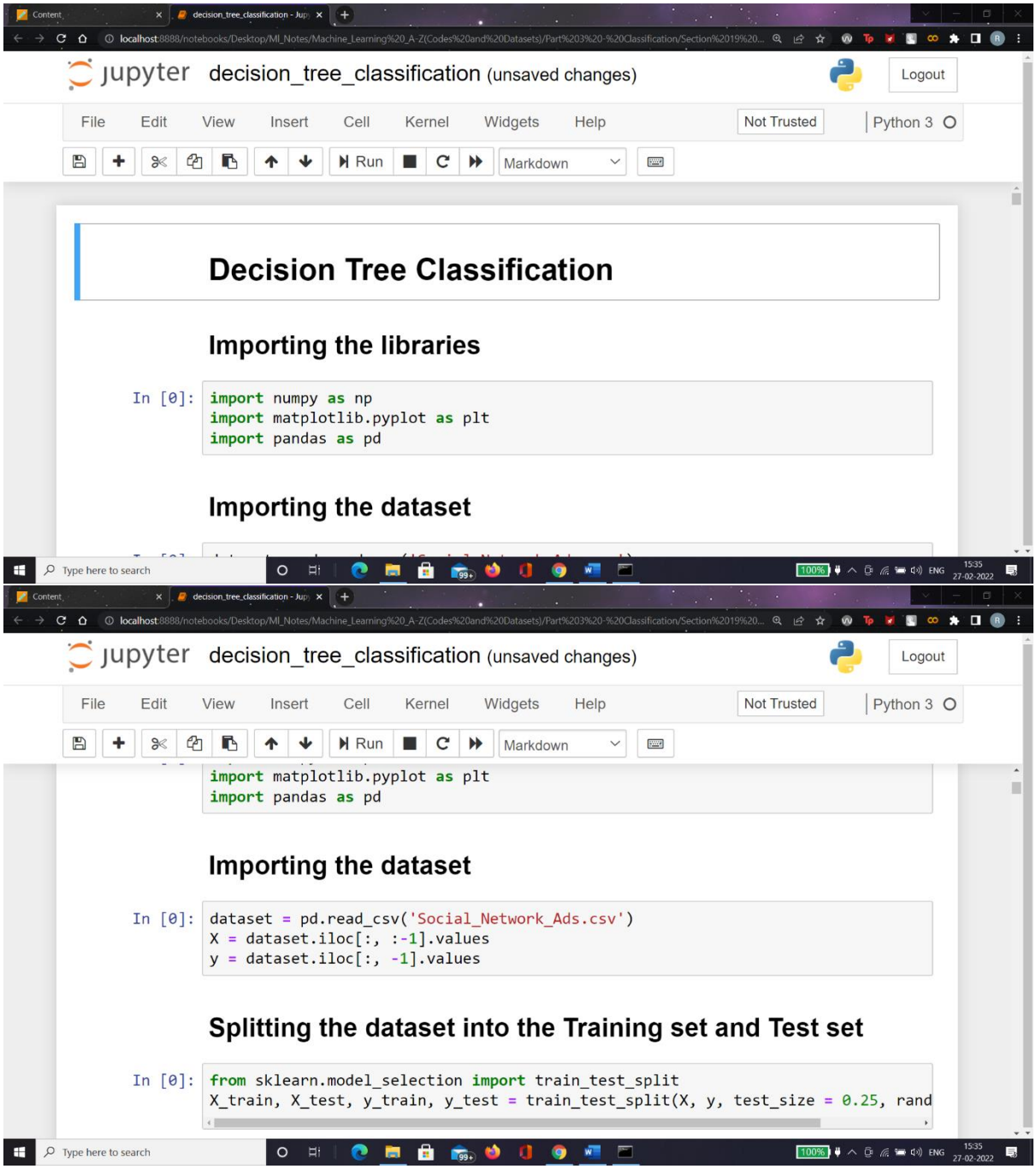
Rohan Nyati

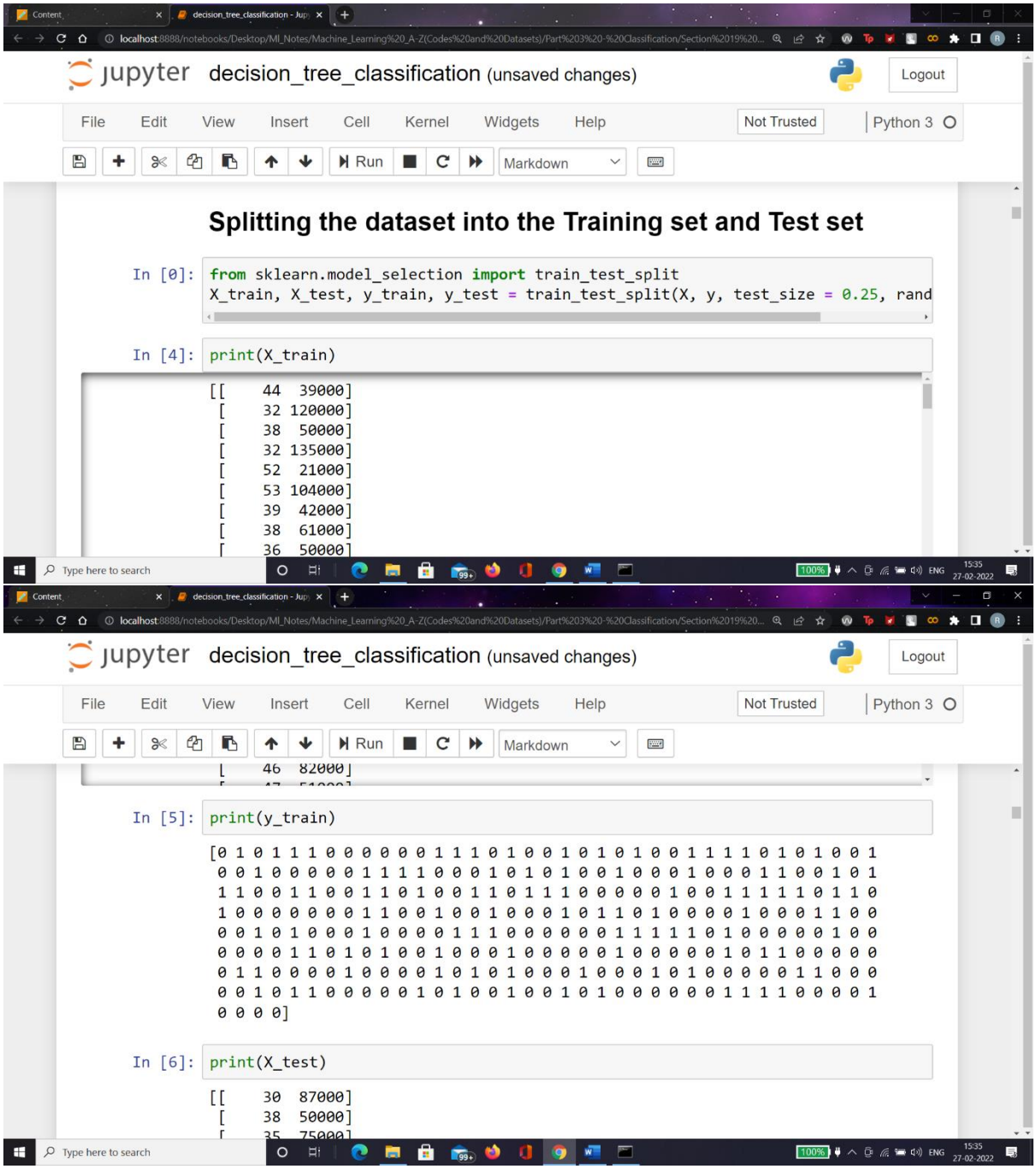
500075940

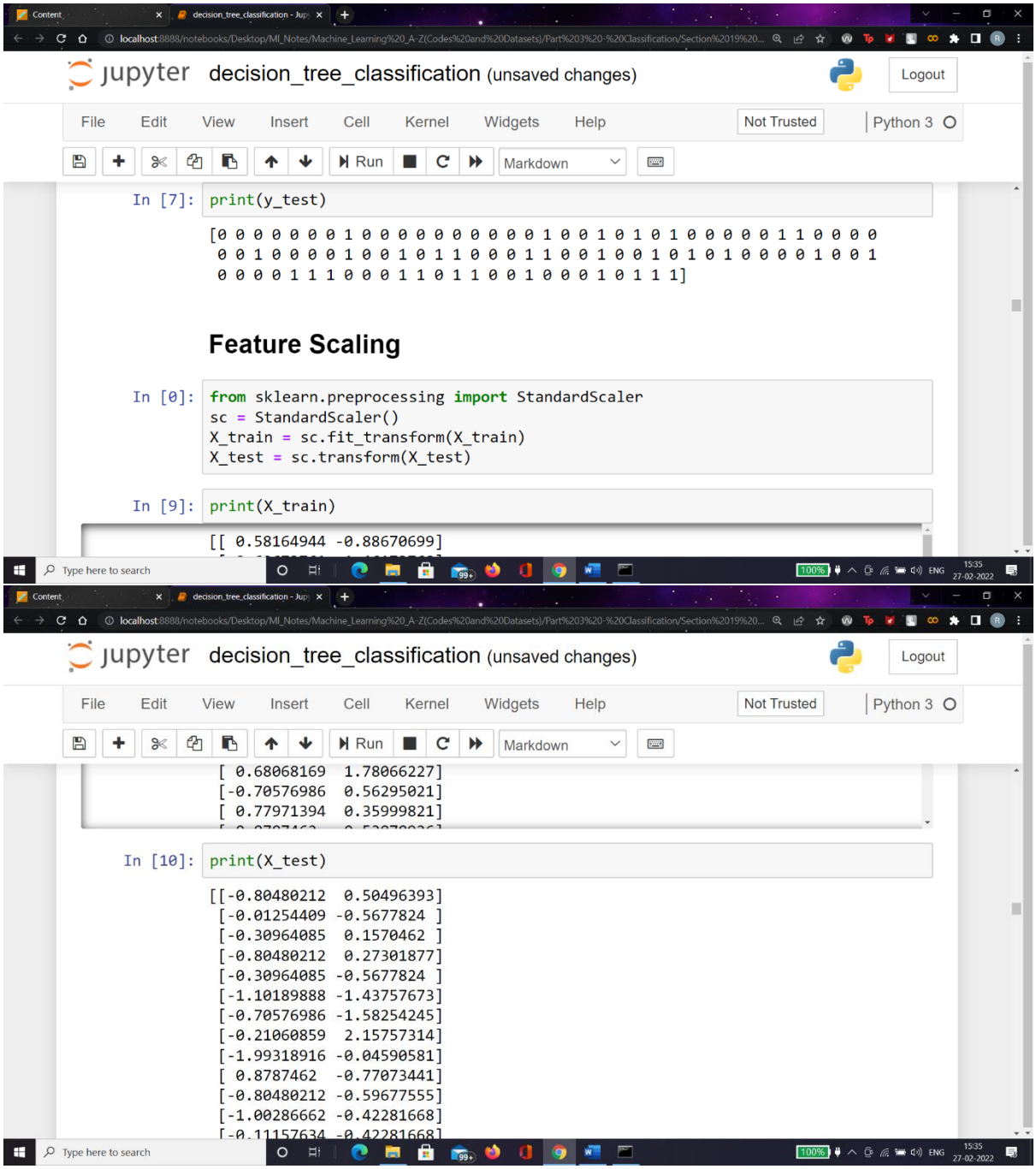
R177219148

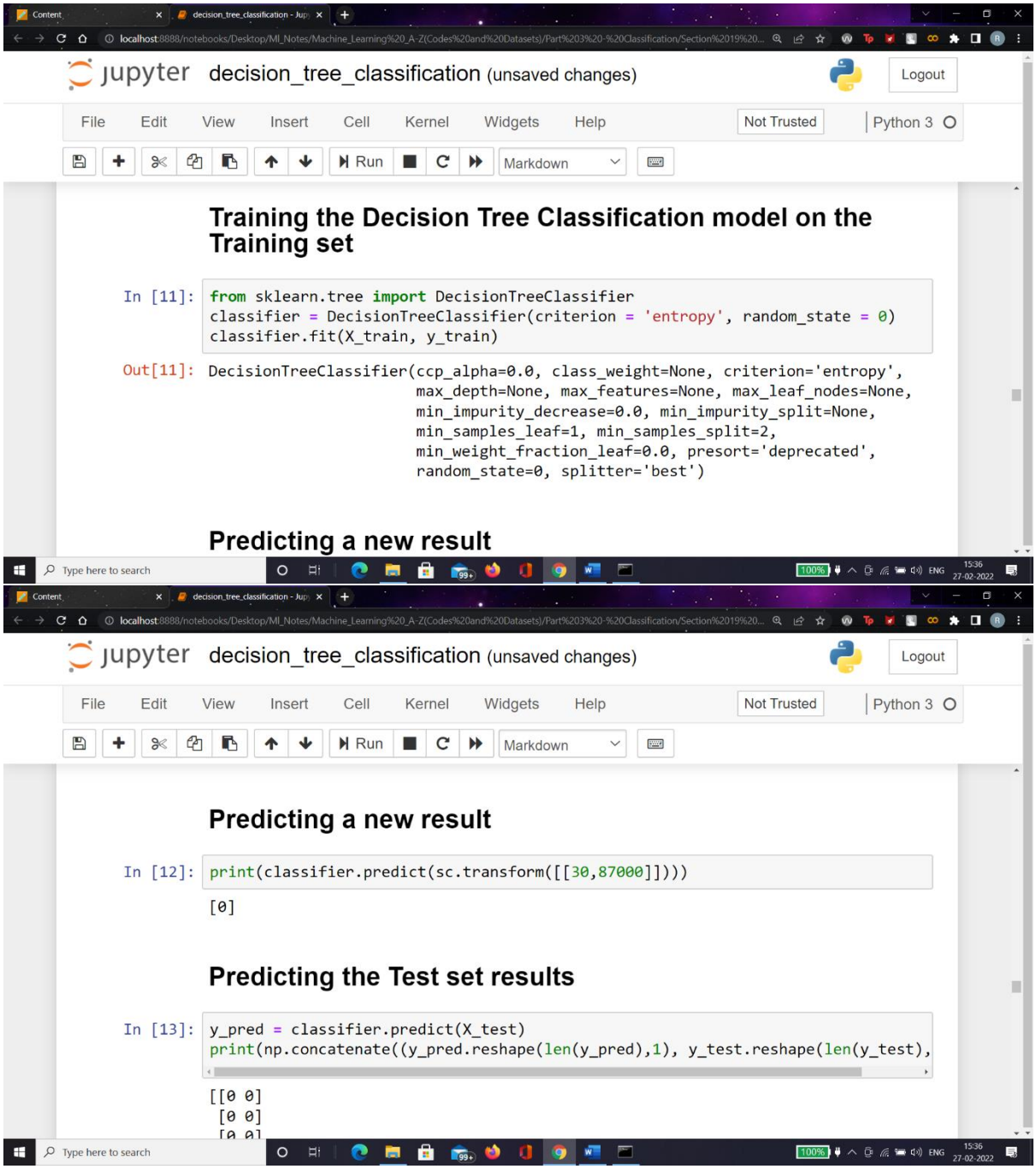
B-5 Ai & Ml

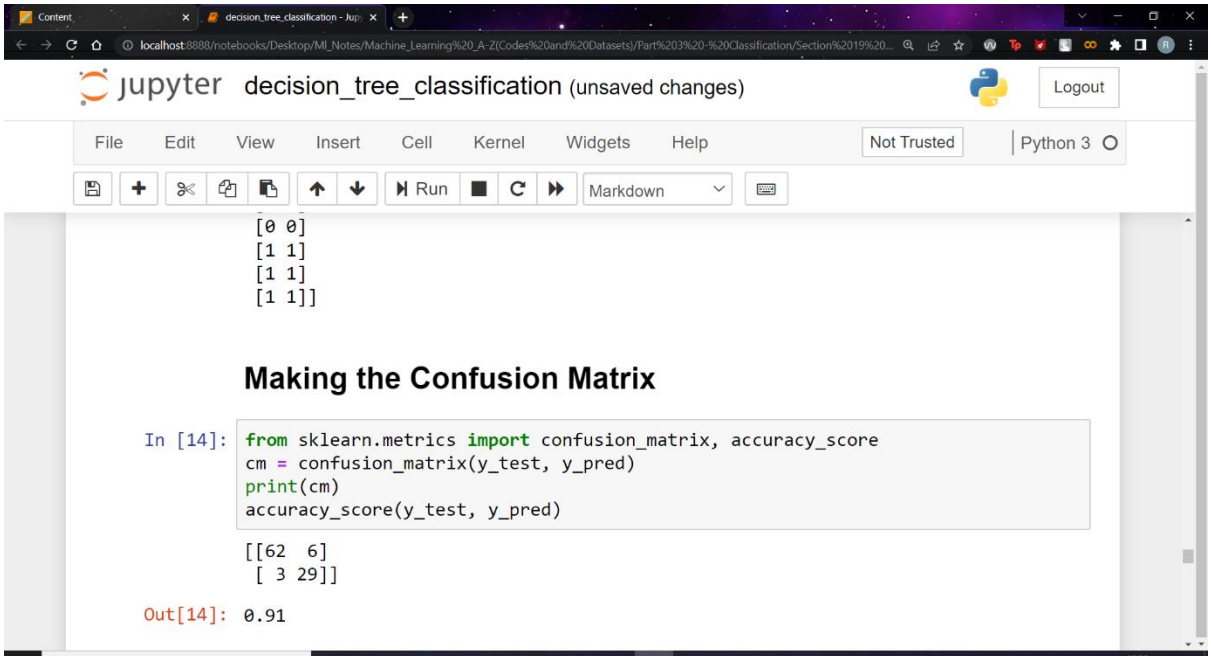
Experiment – 5

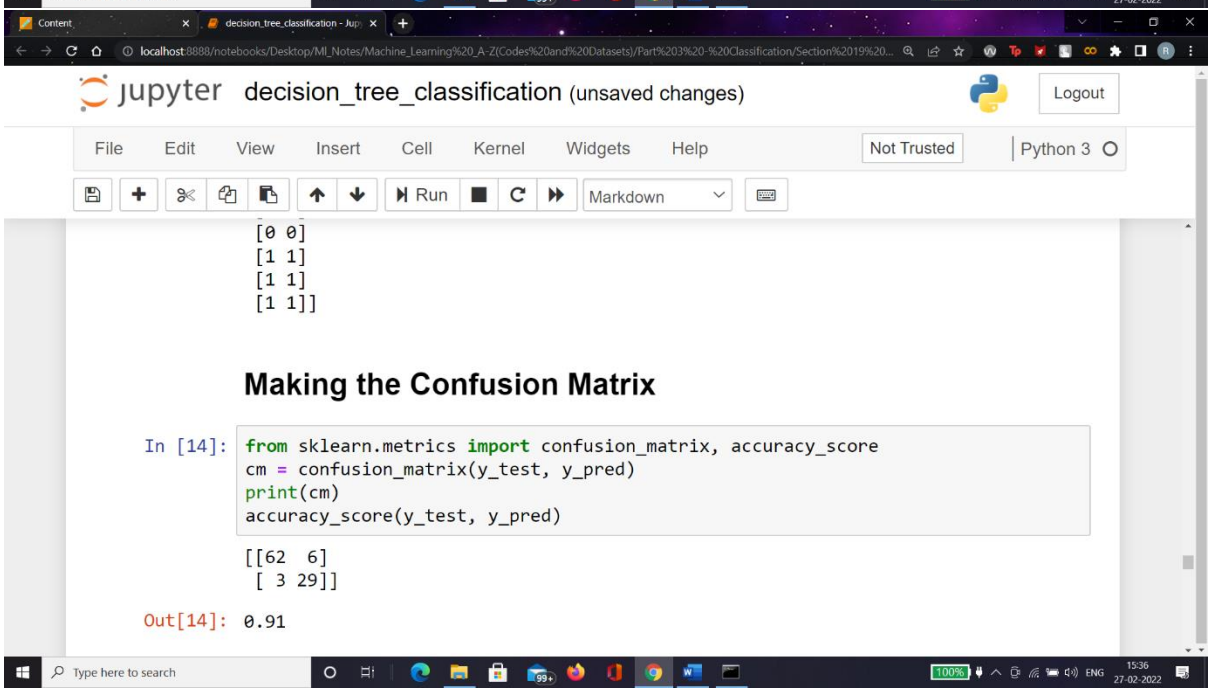



jupyter decision_tree_classification (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
Splitting the dataset into the Training set and Test set
In [0]:
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, rand
In [4]:
print(X_train)
[[44 39000]
[32 120000]
[38 50000]
[32 135000]
[52 21000]
[53 104000]
[39 42000]
[38 61000]
[36 50000]
In [5]:
print(y_train)
[0 1 0 1 1 1 0 0 0 0 0 0 1 1 1 0 1 0 0 1 0 1 0 1 0 0 1 1 1 1 0 1 0 1 0 0 1
0 0 1 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 1 1 0 0 1 0 1
1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 1 1 1 0 0 0 0 0 1 0 0 1 1 1 1 1 1 0 1 1 0
1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0 1 0 0 0 1 1 0 0
0 0 1 0 1 0 0 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 0 0 1 0 0
0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 0 0 0 0 1 0 0 0 0 0 1 0 1 1 0 0 0 0 0
0 1 1 0 0 0 0 1 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0 1 1 0 0 0
0 0 1 0 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1
0 0 0 0]
In [6]:
print(X_test)
[[30 87000]
[38 50000]


jupyter decision_tree_classification (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
In [7]:
print(y_test)
[0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 0 0 0 0 0 1 1 0 0 0 0
0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 0 1 0 0 0 0 1 0 0 1
0 0 0 0 1 1 1 0 0 0 1 1 0 1 1 0 0 1 0 0 0 1 0 1 1 1]
Feature Scaling
In [0]:
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
In [9]:
print(X_train)
[[0.58164944 -0.88670699]
[0.68068169 1.78066227]
[-0.70576986 0.56295021]
[0.77971394 0.35999821]
In [10]:
print(X_test)
[[-0.80480212 0.50496393]
[-0.01254409 -0.5677824]
[-0.30964085 0.1570462]
[-0.80480212 0.27301877]
[-0.30964085 -0.5677824]
[-1.10189888 -1.43757673]
[-0.70576986 -1.58254245]
[-0.21060859 2.15757314]
[-1.99318916 -0.04590581]
[0.8787462 -0.77073441]
[-0.80480212 -0.59677555]
[-1.00286662 -0.42281668]


jupyter decision_tree_classification (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
Training the Decision Tree Classification model on the Training set
In [11]:
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy', random_state = 0)
classifier.fit(X_train, y_train)
Out[11]:
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='entropy',
max_depth=None, max_features=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None,
min_samples_leaf=1, min_samples_split=2,
min_weight_fraction_leaf=0.0, presort='deprecated',
random_state=0, splitter='best')
Predicting a new result
Predicting a new result
In [12]:
print(classifier.predict(sc.transform([[30,87000]])))
[0]
Predicting the Test set results
In [13]:
y_pred = classifier.predict(X_test)
print(np.concatenate((y_pred.reshape(len(y_pred),1), y_test.reshape(len(y_test),
[[0 0]
[0 0]


Content
decision_tree_classification - Jup
localhost:8888/notebooks/Desktop/ML_Notes/Machine_Learning%20_A-Z(Codes%20and%20Datasets)/Part%203%20-%20Classification/Section%2019%20...
jupyter decision_tree_classification (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
[0 0]
[1 1]
[1 1]
[1 1]]
Making the Confusion Matrix
In [14]:
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
[[62 6]
[3 29]]
Out[14]: 0.91
Type here to search
100%
ENG
15:36
27-02-2022


Content
decision_tree_classification - Jup
localhost:8888/notebooks/Desktop/ML_Notes/Machine_Learning%20_A-Z(Codes%20and%20Datasets)/Part%203%20-%20Classification/Section%2019%20...
jupyter decision_tree_classification (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Markdown
[0 0]
[1 1]
[1 1]
[1 1]]
Making the Confusion Matrix
In [14]:
from sklearn.metrics import confusion_matrix, accuracy_score
cm = confusion_matrix(y_test, y_pred)
print(cm)
accuracy_score(y_test, y_pred)
[[62 6]
[3 29]]
Out[14]: 0.91
Type here to search
100%
ENG
15:36
27-02-2022

jupyter decision_tree_classification (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown

In [15]:
```
from matplotlib.colors import ListedColormap
X_set, y_set = sc.inverse_transform(X_train), y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 10, stop = X_set[:, 0
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:,
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ra
             alpha = 0.75, 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], c = ListedColormap((
plt.title('Decision Tree Classification (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be
avoided as value-mapping will have precedence in case its length matches with
```

jupyter decision_tree_classification (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown

```
ify the same RGB or RGBA value for all points.
```

jupyter decision_tree_classification (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown

Visualising the Test set results

In [16]:
```
from matplotlib.colors import ListedColormap
X_set, y_set = sc.inverse_transform(X_test), y_test
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 10, stop = X_set[:, 0
                     np.arange(start = X_set[:, 1].min() - 1000, stop = X_set[:,
plt.contourf(X1, X2, classifier.predict(sc.transform(np.array([X1.ravel(), X2.ra
             alpha = 0.75, 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], c = ListedColormap((
plt.title('Decision Tree Classification (Test set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

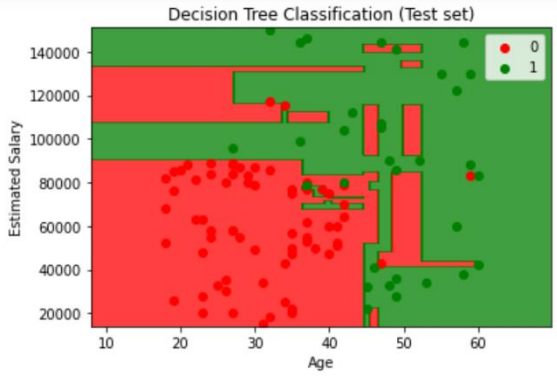
Type here to search

jupyter decision_tree_classification (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Markdown



Type here to search