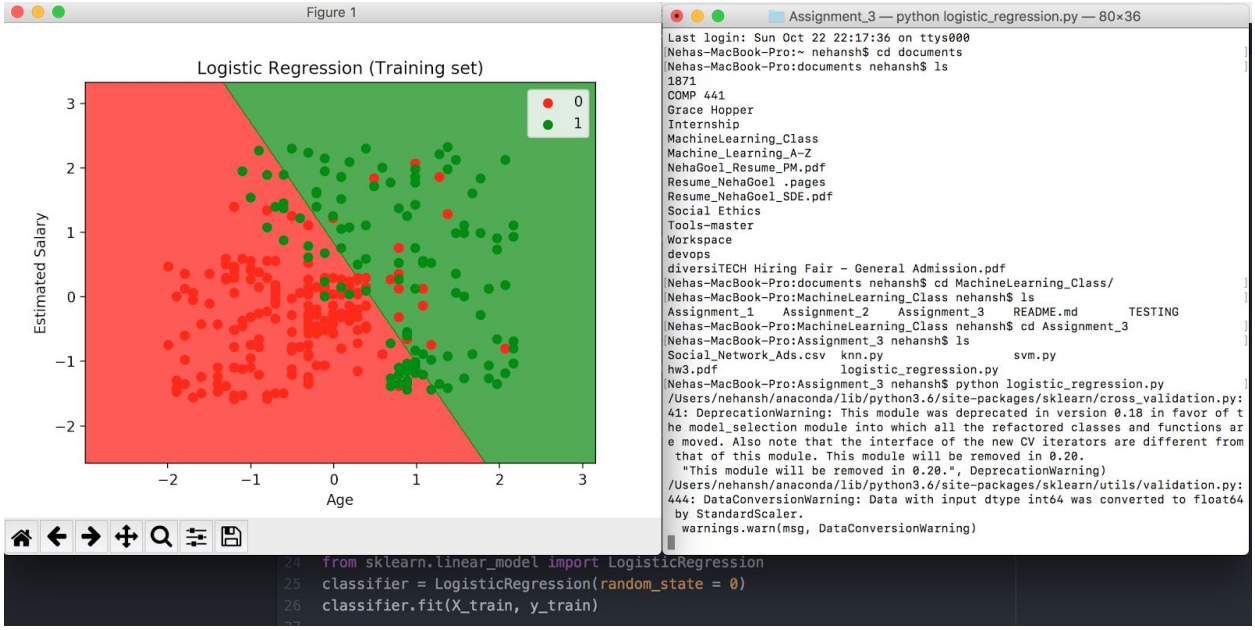
The dataset I used for this homework, is a dataset of Social Network Ads, with parameters of User ID,Gender,Age,Estimated Salary,Purchased. Here with different algorithms I am predicting based on the estimated salary and the Age of the user whether, that person will buy a SUV or not. In all my graphs below Green Dots & Area (1) will represent people who bought SUV and RED area as the people who did not bought SUV.

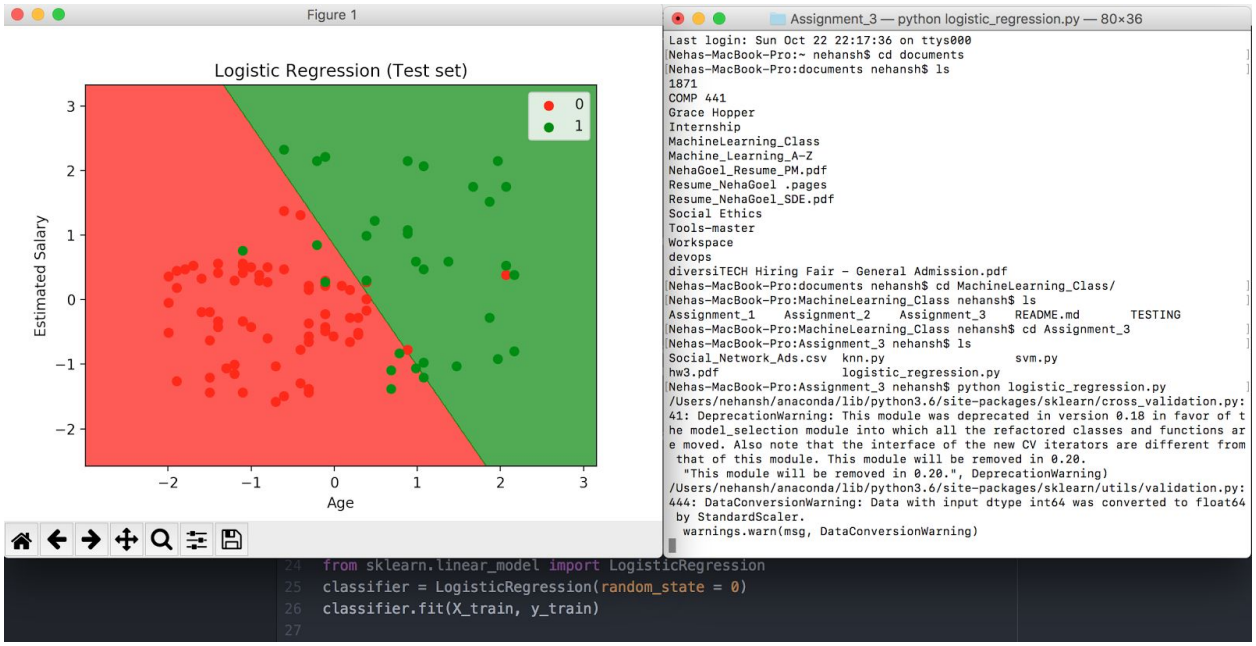
ANSWER 1

Logistic Regression Implementation in sckit-learn with default parameters.
As a result of logistic regression, we can see in graph that the algorithm predicted few green dots(people who bought SUV) as the people who did not bought SUV

1. On Training Set :



2. On test Set:



SVM Implementation in scki-learn

1. For Training Set:

2. For Test Set:

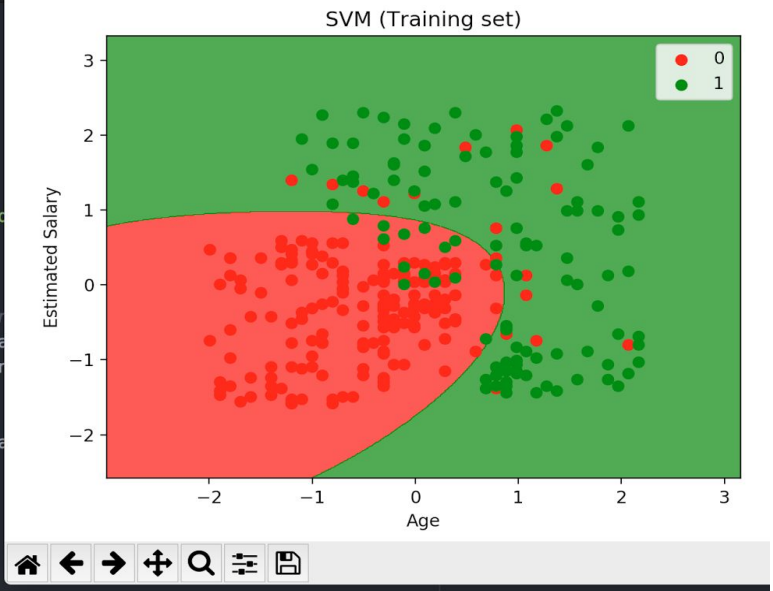
ANSWER 2:

1. Tweaking the parameters of SVM:

classifier = SVC(kernel = 'rbf', random_state = 0, gamma = 0.10, C=10.0)
With these parameters, It proved to be the best SVM performance with minimal incorrect prediction. Accuracy score of 0.92 (showed in Answer 4)

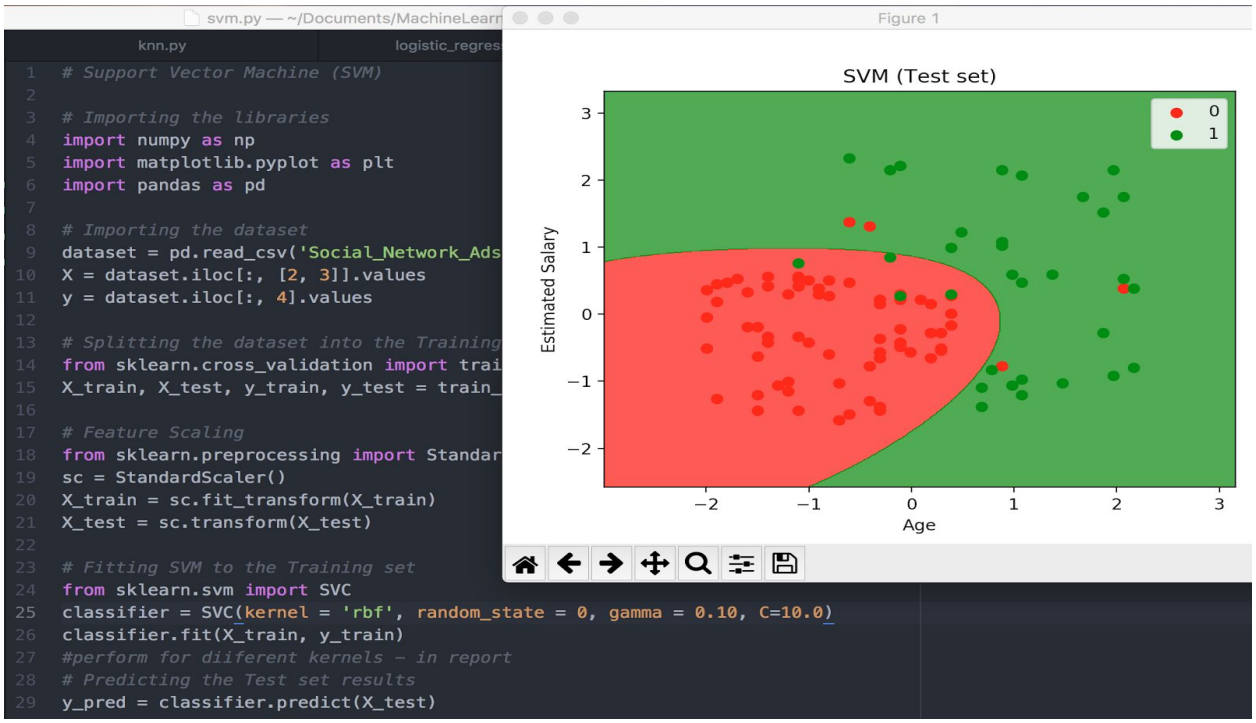
Training Set:

```python
# Support Vector Machine (SVM)

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Network_Ad
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Splitting the dataset into the Trainin
from sklearn.cross_validation import tra
X_train, X_test, y_train, y_test = train

# Feature Scaling
from sklearn.preprocessing import Standa
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting SVM to the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'rbf', random_state = 0, gamma = 0.10, C=10.0)
classifier.fit(X_train, y_train)
#perform for diiferent kernels - in report
# Predicting the Test set results
y_pred = classifier.predict(X_test)
```



TEST SET:

It Improved the performance in comparison to previous implementation on test set. Very few incorrect predictions in Test Set

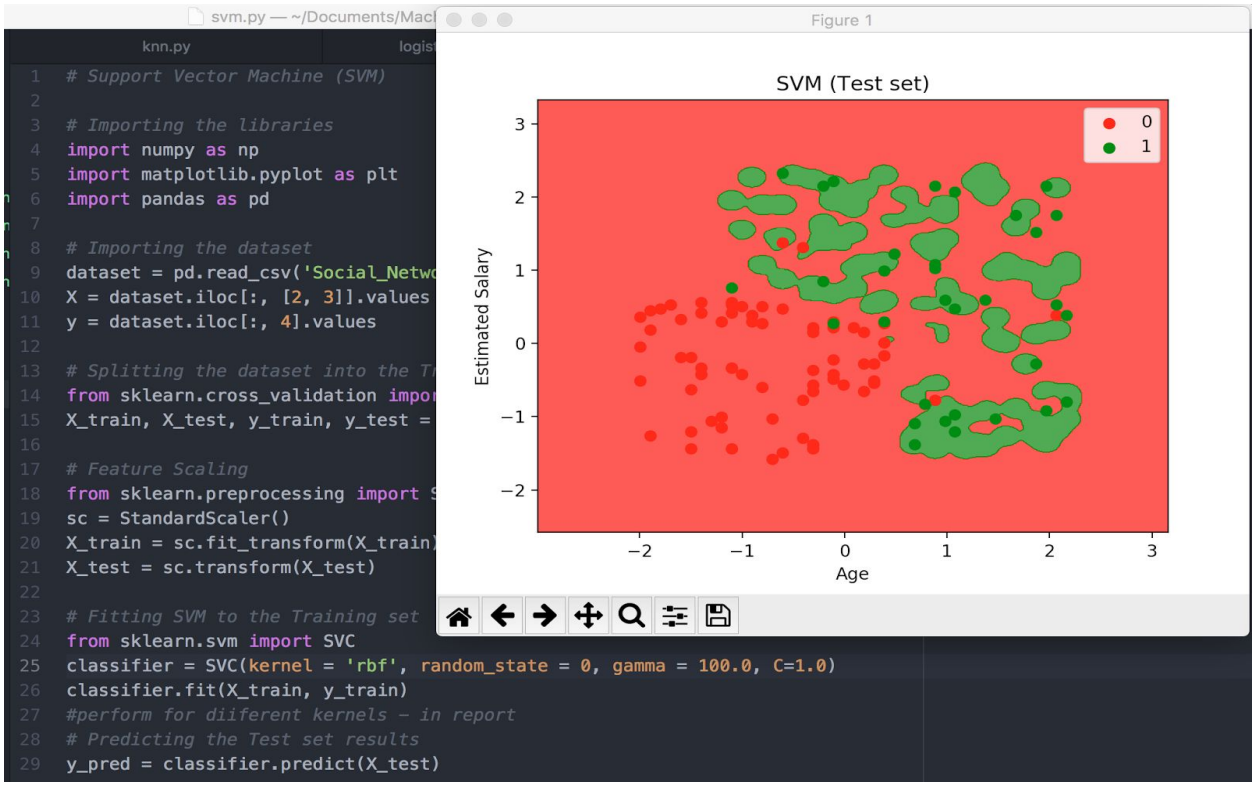


Tweaking parameters example 2:

classifier = SVC(kernel = 'rbf', random_state = 0, gamma = 100.0, C=1.0)

Training Set:

TEST SET:



ANSWER 3 :

On performing KNN classifier on the data set I got a score of 0.93 with just 3 to 4 incorrect predictions in the test set. It outperforms among all the algorithms.

KNN Classifier: for Training Set


knn.py — ~/Documents/MachineLear
knn.py
logistic_regre
# K-Nearest Neighbors (K-NN)
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
# Importing the dataset
dataset = pd.read_csv('Social_Network_Ad
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values
# Splitting the dataset into the Training
from sklearn.cross_validation import tra
X_train, X_test, y_train, y_test = train_
# Feature Scaling
from sklearn.preprocessing import Standa
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
# Fitting K-NN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)
# Predicting the Test set results
y_pred = classifier.predict(X_test)
Figure 1
K-NN (Training set)
Estimated Salary
Age
0
1
3
2
1
0
−1
−2
−2
−1
0
1
2
3

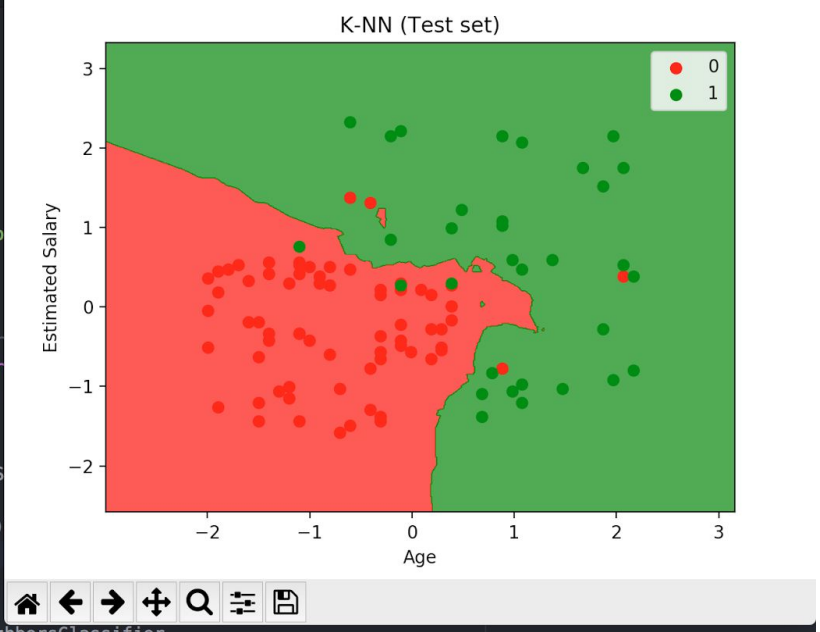
KNN - Test Set

knn.py — ~/Documents/Mach

knn.py

```python
# K-Nearest Neighbors (K-NN)

# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Social_Netwo
X = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values

# Splitting the dataset into the Tr
from sklearn.cross_validation impor
X_train, X_test, y_train, y_test =

# Feature Scaling
from sklearn.preprocessing import S
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

# Fitting K-NN to the Training set
from sklearn.neighbors import KNeighborsClassifier
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)
classifier.fit(X_train, y_train)

# Predicting the Test set results
y_pred = classifier.predict(X_test)
```

Figure 1



ANSWER 4

Here I compare the classification report for SVM various models and the KNN:

from sklearn.metrics import classification_report
print (classification_report (y_test, y_pred))

SVM Linear

```
Nehas-MacBook-Pro:Assignment_3 nehansh$ python svm.py
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/cross_validation.py:
41: DeprecationWarning: This module was deprecated in version 0.18 in favor of t
he model_selection module into which all the refactored classes and functions ar
e moved. Also note that the interface of the new CV iterators are different from
 that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/utils/validation.py:
444: DataConversionWarning: Data with input dtype int64 was converted to float64
 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
             precision    recall  f1-score   support

          0       0.89      0.97      0.93        68
          1       0.92      0.75      0.83        32

avg / total       0.90      0.90      0.90       100
```

SVM RBF (kernel = 'rbf', random_state = 0, gamma = 0.10, C=10.0)

```
Assignment_3 — python svm.py — 80×24
Nehas-MacBook-Pro:Assignment_3 nehansh$ python svm.py
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/cross_validation.py:
41: DeprecationWarning: This module was deprecated in version 0.18 in favor of t
he model_selection module into which all the refactored classes and functions ar
e moved. Also note that the interface of the new CV iterators are different from
 that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/utils/validation.py:
444: DataConversionWarning: Data with input dtype int64 was converted to float64
 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
             precision    recall  f1-score   support

          0       0.94      0.94      0.94        68
          1       0.88      0.88      0.88        32

avg / total       0.92      0.92      0.92       100
```

SVM - (kernel = 'rbf', random_state = 0, gamma = 100.0, C=1.0)

```
Nehas-MacBook-Pro:Assignment_3 nehansh$ python svm.py
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/cross_validation.py:
41: DeprecationWarning: This module was deprecated in version 0.18 in favor of t
he model_selection module into which all the refactored classes and functions ar
e moved. Also note that the interface of the new CV iterators are different from
 that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/utils/validation.py:
444: DataConversionWarning: Data with input dtype int64 was converted to float64
 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
             precision    recall  f1-score   support

          0       0.80      0.97      0.88        68
          1       0.89      0.50      0.64        32

avg / total       0.83      0.82      0.80       100
```

KNN

```
Nehas-MacBook-Pro:Assignment_3 nehansh$ python knn.py
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/cross_validation.py:
41: DeprecationWarning: This module was deprecated in version 0.18 in favor of t
he model_selection module into which all the refactored classes and functions ar
e moved. Also note that the interface of the new CV iterators are different from
 that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
/Users/nehansh/anaconda/lib/python3.6/site-packages/sklearn/utils/validation.py:
444: DataConversionWarning: Data with input dtype int64 was converted to float64
 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
[[64  4]
 [ 3 29]]
             precision    recall  f1-score   support

          0       0.96      0.94      0.95        68
          1       0.88      0.91      0.89        32

avg / total       0.93      0.93      0.93       100
```

The best SVM was when kernel is rbf, gamma = 0.10, C=10.0. On comparing it with KNN, KNN performed slightly better than it.