QM-7063 Data Mining
Professor: Dr. Abdulrashid
Learning Practice 9 – Noah L. Schrick

Using the accidentsFull.csv dataset, perform the following tasks:

a.) Numerize the dataset
The dataset primarily contained integer values, however assurance was needed to ensure that the 0/1 data values were not interpreted as categorical values by pandas. In addition, the 'Injury' column needed to be added. This column could be filled with string values ('yes'/'no') or 0/1 values. In either case, the data needed to be converted or ensured to be numerical.

This was performed by adding the 'Injury' column and adding values as 0 and 1. Since all columns had integer values at this point, all columns were converted to numerical using the pandas library.

b.) Transform the data by either normalizing or standardizing it
Data was normalized using scikit-learn's pre-processing library. The standard scalar function was used to transform the data (after removing the Injury column since it is the outcome variable).
After the data was normalized, it was concatenated together with the outcome variable (Injury) to form a new, normalized data frame. From this normalized data frame, the location of the train data and validation data was identified using iloc.

c.) Use train, test, and split function to split the data into training and testing sets
Data was partitioned using scikit-learn's model selection train_test_split function for a 60/40 split with random state set to 20.

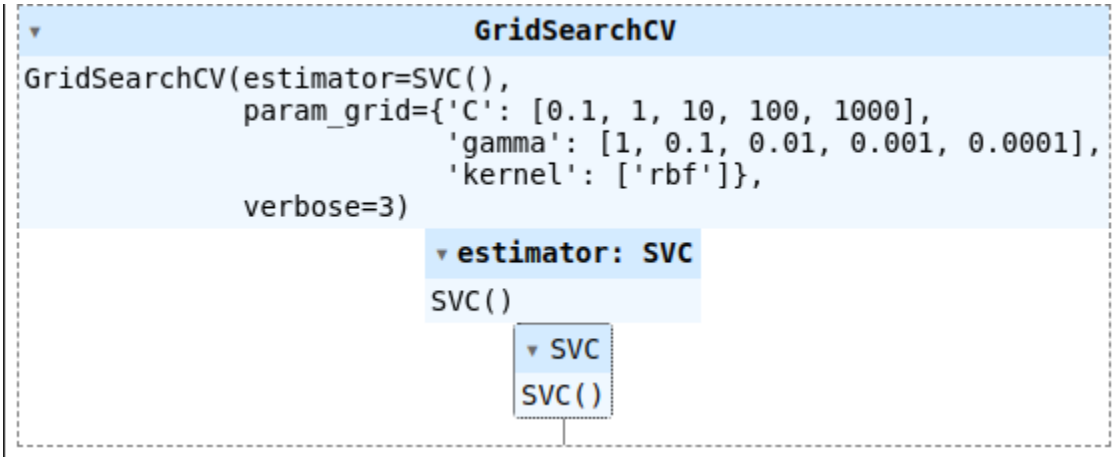
d.)Select your preferred kernel type and determine the kernel values by using either grid-search or v-fold cross validation.
GridSearchCV with an underling call to SVC() with the rbf kernel was the approach for this task. A grid parameter was first created with regularization parameter options of 0.1, 1.0, 10, 100, and 1000. The kernel coefficient options for rbf were 0.0001, 0.001, 0.01, 0.1, and 1.0. For identifying optimal parameters, GridSearchCV scored each option. With the v-fold cross-validation, and 25 parameter permutation options, a total of 125 fits were performed.

To prevent running GridSearchCV a second time after optimal parameters were selected, the 'refit' option was set to 'True'. This allows the grid estimator to automatically be refit with the best parameters found.

For this data, the optimal regularization parameter was determined to be 0.1, and the optimal kernel coefficient was determined to be 0.001.

e.) Run a SVM classifier using identified kernel values found in (d).

Since the grid was automatically refit with the parameters found previously, the grid just needed to have the predict function called with the data to create the grid predictions. The predict function returned the predicted labels or values.



f.) Obtain the confusion matrix.

The confusion matrix was obtained using the validation set and the grid predictions. The confusion matrix is shown below.

Actual/Predicted	**Positive**	**Negative**
Positive	8270	0
Negative	0	8604

g.) What is the overall error for the validation set?

	Precision	**Recall**	**F1-Score**	**Support**
0	1.00	1.00	1.00	8270
1	1.00	1.00	1.00	8604
Accuracy			1.00	16874
Macro Average	1.0	1.0	1.0	16874
Weighted Average	1.0	1.0	1.0	16874

The SVM scored with perfect accuracy, which immediately brings about concerns. Though a high accuracy may be expected, perfect accuracy raises a few flags. In this case, the data is easily able to be captured, with or without SVM. When plotting the data, it is apparent that a curve can be drawn for the outcome variable Injury. For example, low speed crashes do not result in Injury=1, and samples such as this make predictions simplistic. Feature importance and the shape of the data indicate that this data will relatively easily score with high accuracy, especially when using v-fold cross validation and tuning the parameters of the grid search, which is arguably overkill for this set of data. Using other data sets, such as SKLearn's breast cancer data set, shows that the SVM will not score perfectly, but will score highly. When looking at the GridSearchCV() output, it is seen that certain parameters resulted in accuracy scores in the 70% range. The best parameters in this case after tuning yielded trained grids scoring 100%.

APPENDIX

CODE

```
# Learning Practice 9 for the University of Tulsa's QM-7063 Data Mining Course
# Support Vector Machines
# Professor: Dr. Abdulrashid, Spring 2023
# Noah L. Schrick – 1492657

# Imports
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report,confusion_matrix

%matplotlib inline

# a
accidents_df = pd.read_csv('accidentsFull.csv')
accidents_df['Injury'] = (accidents_df['MAX_SEV_IR'] > 0).astype(int)
accidents_df = accidents_df.apply(pd.to_numeric) # convert all columns of DataFrame

# b. and c.
scaler = preprocessing.StandardScaler()

accident_cols = accidents_df.columns.values.tolist()
accident_cols.remove('Injury')

# split into training and validation
trainData, validData = train_test_split(accidents_df, test_size=0.40, random_state=20)

scaler.fit(trainData[accident_cols])  # Note the use of an array of column names

# Transform the full dataset
accidentNorm = pd.concat([pd.DataFrame(scaler.transform(accidents_df[accident_cols]),
                                       columns=accident_cols),
                          accidents_df[['Injury']]], axis=1)

trainNorm = accidentNorm.iloc[trainData.index]
validNorm = accidentNorm.iloc[validData.index]

# c.
train_X = trainNorm[accident_cols]
train_y = trainNorm['Injury']
valid_X = validNorm[accident_cols]
```

```
valid_y = validNorm['Injury']

# d.
param_grid = {'C': [0.1,1, 10, 100, 1000], 'gamma': [1,0.1,0.01,0.001,0.0001], 'kernel': ['rbf']}
grid = GridSearchCV(SVC(),param_grid,refit=True,verbose=3)
grid.fit(train_X,train_y)

# Best options:
grid.best_params_
print()
grid.best_estimator_

# e.
grid_predictions = grid.predict(valid_X)

# f.
print(confusion_matrix(valid_y,grid_predictions))

# g.
print(classification_report(valid_y,grid_predictions))
```

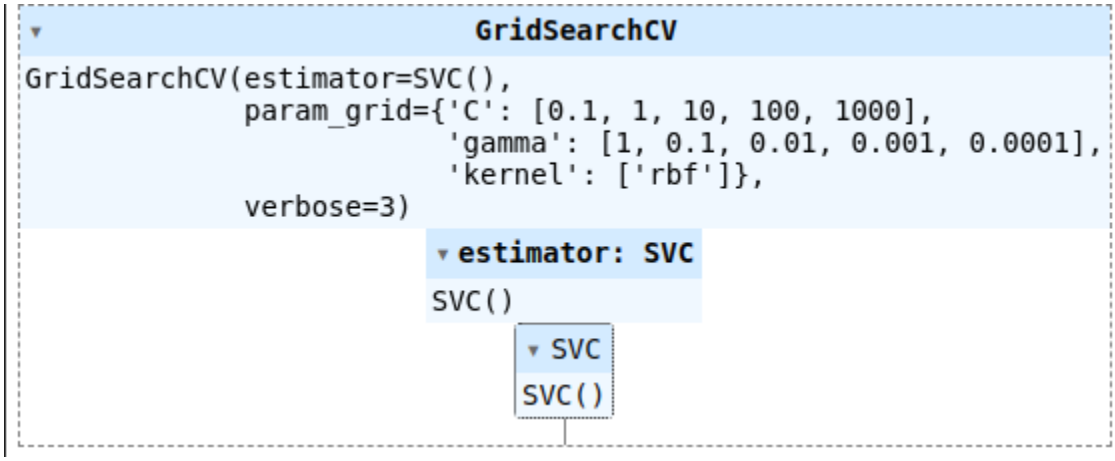
OUTPUTS

```
Fitting 5 folds for each of 25 candidates, totalling 125 fits
[CV 1/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.772 total time=  33.9s
[CV 2/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.762 total time=  46.2s
[CV 3/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.769 total time=  42.3s
[CV 4/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.762 total time=  36.2s
[CV 5/5] END ........C=0.1, gamma=1, kernel=rbf;, score=0.761 total time=  34.8s
[CV 1/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.999 total time=   5.6s
[CV 2/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.996 total time=   5.8s
[CV 3/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.998 total time=   5.6s
[CV 4/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.995 total time=   5.6s
[CV 5/5] END ......C=0.1, gamma=0.1, kernel=rbf;, score=0.997 total time=   5.7s
[CV 1/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=1.000 total time=   1.1s
[CV 2/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=1.000 total time=   1.1s
[CV 3/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=1.000 total time=   1.2s
[CV 4/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=0.999 total time=   1.4s
[CV 5/5] END .....C=0.1, gamma=0.01, kernel=rbf;, score=1.000 total time=   1.0s
[CV 1/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=1.000 total time=   4.4s
[CV 2/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=1.000 total time=   4.3s
[CV 3/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=1.000 total time=   4.2s
[CV 4/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=1.000 total time=   4.3s
[CV 5/5] END ....C=0.1, gamma=0.001, kernel=rbf;, score=1.000 total time=   4.4s
[CV 1/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=1.000 total time=  26.5s
[CV 2/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=1.000 total time=  25.9s
[CV 3/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=1.000 total time=  27.3s
[CV 4/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=1.000 total time=  25.9s
[CV 5/5] END ...C=0.1, gamma=0.0001, kernel=rbf;, score=1.000 total time=  26.6s
```

[CV 1/5] END .....C=1, gamma=1, kernel=rbf;, score=0.923 total time= 1.1min
[CV 2/5] END .....C=1, gamma=1, kernel=rbf;, score=0.913 total time= 1.3min
[CV 3/5] END .....C=1, gamma=1, kernel=rbf;, score=0.917 total time= 1.1min
[CV 4/5] END .....C=1, gamma=1, kernel=rbf;, score=0.916 total time= 1.1min
[CV 5/5] END .....C=1, gamma=1, kernel=rbf;, score=0.912 total time= 1.1min
[CV 1/5] END .....C=1, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.4s
[CV 2/5] END .....C=1, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.3s
[CV 3/5] END .....C=1, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.4s
[CV 4/5] END .....C=1, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.3s
[CV 5/5] END .....C=1, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.4s
[CV 1/5] END .....C=1, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 2/5] END .....C=1, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 3/5] END .....C=1, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 4/5] END .....C=1, gamma=0.01, kernel=rbf;, score=0.999 total time= 0.4s
[CV 5/5] END .....C=1, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 1/5] END .....C=1, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.6s
[CV 2/5] END .....C=1, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.6s
[CV 3/5] END .....C=1, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.6s
[CV 4/5] END .....C=1, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.6s
[CV 5/5] END .....C=1, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.7s
[CV 1/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=1.000 total time= 4.6s
[CV 2/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=1.000 total time= 4.9s
[CV 3/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=1.000 total time= 4.4s
[CV 4/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=1.000 total time= 5.0s
[CV 5/5] END .....C=1, gamma=0.0001, kernel=rbf;, score=1.000 total time= 4.9s
[CV 1/5] END .....C=10, gamma=1, kernel=rbf;, score=0.928 total time= 1.1min
[CV 2/5] END .....C=10, gamma=1, kernel=rbf;, score=0.918 total time= 1.2min
[CV 3/5] END .....C=10, gamma=1, kernel=rbf;, score=0.923 total time= 1.3min
[CV 4/5] END .....C=10, gamma=1, kernel=rbf;, score=0.920 total time= 1.0min
[CV 5/5] END .....C=10, gamma=1, kernel=rbf;, score=0.918 total time= 1.1min
[CV 1/5] END .....C=10, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.2s
[CV 2/5] END .....C=10, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.1s
[CV 3/5] END .....C=10, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.0s
[CV 4/5] END .....C=10, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.1s
[CV 5/5] END .....C=10, gamma=0.1, kernel=rbf;, score=1.000 total time= 4.1s
[CV 1/5] END .....C=10, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.5s
[CV 2/5] END .....C=10, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.5s
[CV 3/5] END .....C=10, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 4/5] END .....C=10, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.5s
[CV 5/5] END .....C=10, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 1/5] END .....C=10, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 2/5] END .....C=10, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 3/5] END .....C=10, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 4/5] END .....C=10, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 5/5] END .....C=10, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 1/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=1.000 total time= 1.0s
[CV 2/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=1.000 total time= 1.0s
[CV 3/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=1.000 total time= 1.0s
[CV 4/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.8s

[CV 5/5] END ....C=10, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.9s
[CV 1/5] END .....C=100, gamma=1, kernel=rbf;, score=0.928 total time= 1.1min
[CV 2/5] END .....C=100, gamma=1, kernel=rbf;, score=0.918 total time= 1.1min
[CV 3/5] END .....C=100, gamma=1, kernel=rbf;, score=0.923 total time= 1.1min
[CV 4/5] END .....C=100, gamma=1, kernel=rbf;, score=0.920 total time= 59.7s
[CV 5/5] END .....C=100, gamma=1, kernel=rbf;, score=0.918 total time= 1.0min
[CV 1/5] END .....C=100, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.2s
[CV 2/5] END .....C=100, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.0s
[CV 3/5] END .....C=100, gamma=0.1, kernel=rbf;, score=0.999 total time= 3.1s
[CV 4/5] END .....C=100, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.1s
[CV 5/5] END .....C=100, gamma=0.1, kernel=rbf;, score=1.000 total time= 3.2s
[CV 1/5] END .....C=100, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 2/5] END .....C=100, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 3/5] END .....C=100, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 4/5] END .....C=100, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 5/5] END .....C=100, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 1/5] END ....C=100, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 2/5] END ....C=100, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 3/5] END ....C=100, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 4/5] END ....C=100, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 5/5] END ....C=100, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 1/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 2/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 3/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 4/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 5/5] END ...C=100, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.5s
[CV 1/5] END .....C=1000, gamma=1, kernel=rbf;, score=0.928 total time= 57.9s
[CV 2/5] END .....C=1000, gamma=1, kernel=rbf;, score=0.918 total time= 1.1min
[CV 3/5] END .....C=1000, gamma=1, kernel=rbf;, score=0.923 total time= 1.1min
[CV 4/5] END .....C=1000, gamma=1, kernel=rbf;, score=0.920 total time= 58.9s
[CV 5/5] END .....C=1000, gamma=1, kernel=rbf;, score=0.918 total time= 59.8s
[CV 1/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=1.000 total time= 2.8s
[CV 2/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.999 total time= 2.7s
[CV 3/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=0.999 total time= 2.6s
[CV 4/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=1.000 total time= 2.7s
[CV 5/5] END .....C=1000, gamma=0.1, kernel=rbf;, score=1.000 total time= 2.6s
[CV 1/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 2/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 3/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.3s
[CV 4/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 5/5] END ....C=1000, gamma=0.01, kernel=rbf;, score=1.000 total time= 0.4s
[CV 1/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 2/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 3/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 4/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.2s
[CV 5/5] END ...C=1000, gamma=0.001, kernel=rbf;, score=1.000 total time= 0.3s
[CV 1/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 2/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 3/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.3s

[CV 4/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.4s
[CV 5/5] END ..C=1000, gamma=0.0001, kernel=rbf;, score=1.000 total time= 0.6s



[[8270 0]
 [0 8604]]

	precision	recall	f1-score	support
0	1.00	1.00	1.00	8270
1	1.00	1.00	1.00	8604
accuracy			1.00	16874
macro avg	1.00	1.00	1.00	16874
weighted avg	1.00	1.00	1.00	16874