Name: Vaibhav Bora

NJIT ucid: vb22

Email Address: vb22@njit.edu

24 November 2024

Professor: Yasser Abduallah

CS 634-101 Data Mining

Final Project Report

Project Technical Overview

For this project, I got my dataset from one of the sources the professor provided which helped me to minimize my time needed to find a good dataset. Also, in the information of the source it mentioned that the datasets had no missing values which helped me to not have to make another algorithm to deal with that problem. The dataset is about direct marketing campaigns from a Portuguese banking institution which were based on different phone calls(sometimes more than one contact for the same person was needed and the reason was to know if the product(i.e. Bank term deposit) was subscribed or not.

Dataset Creation and Management

There were four different datasets which were given from the source but I picked the ‘bank.csv’ dataset because it was randomly selected with a few examples and not all. Moreover, it helped me calculate more computationally demanding algorithms, like SVM, more smoothly. The goal of the experiment was to predict if the clients were subscribed to the product or not using the variable y.

Development and Implementation Tools

The main and primary programming and editing I performed occurred in the Jupyter Notebook, which is a powerful tool that actively supports interactive data science and scientific computing. Jupyter Notebook provided an interactive approach to coding and allowed for immediate feedback and continuous refinement of the python code. For the final verification, I ran the code in IDLE, Python's integrated development and learning environment which is a very important step for ensuring the reliability of my code.

Key Technologies Used

Jupyter Notebook: Served as my main platform for developing, testing, and visualizing the algorithm.

Microsoft Excel: Generated and downloaded all the datasets and CSV files from here.

Python IDLE: Used in the final stages of the project to verify the completeness of the code.

ChatGPT: Used for helping understanding the python libraries and utilized for the verification.

Conclusion

After completing this project, it showed me the very useful applications of different algorithms, mainly Random Forest Classifier, SVM and LSTM. Not only that, learning how to implement them in real life examples and using a real technical dataset to experiment with gave me valuable knowledge and experience.

Screenshots

age;"job";"marital";"education";"default";"balance";"housing";"loan";"contact";"day";"month";"duration";"campaign";"pdays";"previous";"poutcome";"y"
30;"unemployed";"married";"primary";"no";1787;"no";"no";"cellular";19;"oct";79;1;-1;0;"unknown";"no"
33;"services";"married";"secondary";"no";4789;"yes";"yes";"cellular";11;"may";220;1;339;4;"failure";"no"
35;"management";"single";"tertiary";"no";1350;"yes";"no";"cellular";16;"apr";185;1;330;1;"failure";"no"
30;"management";"married";"tertiary";"no";1476;"yes";"yes";"unknown";3;"jun";199;4;-1;0;"unknown";"no"
59;"blue-collar";"married";"secondary";"no";0;"yes";"no";"unknown";5;"may";226;1;-1;0;"unknown";"no"
35;"management";"single";"tertiary";"no";747;"no";"no";"cellular";23;"feb";141;2;176;3;"failure";"no"
36;"self-employed";"married";"tertiary";"no";307;"yes";"no";"cellular";14;"may";341;1;330;2;"other";"no"
39;"technician";"married";"secondary";"no";147;"yes";"no";"cellular";6;"may";151;2;-1;0;"unknown";"no"
41;"entrepreneur";"married";"tertiary";"no";221;"yes";"no";"unknown";14;"may";57;2;-1;0;"unknown";"no"

Figure 1: Bank.csv

```
# Importing all the necessary libraries for the code to run smoothly
import pandas as pd
import numpy as np
from sklearn.model_selection import StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import confusion_matrix, roc_auc_score, brier_score_loss, f1_score
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense, Dropout, Input
from tensorflow.keras.utils import to_categorical
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, f1_score
import time
```

Figure 2: All the libraries necessary to run the program

In order to install these libraries such as pandas or numpy, just type ‘pip install pandas’ or ‘pip install numpy’ respectively in your command prompt.

```
# Defining the LSTM model using input layer alongside input shape
def create_lstm_model(shape_input):
    model = Sequential()
    model.add(Input(shape=shape_input)) # Defining the shape
    model.add(LSTM(50, activation='relu', return_sequences=True)) # Setting return sequence to True
    model.add(Dropout(0.2)) # Drouput helps out with regularization
    model.add(LSTM(20, activation='relu')) # Another layer
    model.add(Dense(1, activation='sigmoid')) # Helping with the binary classification
    model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy']) # Compilation
    return model
```

Figure 3: Creating a model for LSTM

```
# Cross-validation loop for each of the three algorithms
for train_index, test_index in kf.split(X_scaled, y):
    X_train, X_test = X_scaled[train_index], X_scaled[test_index] # Splitting into training and testing sets
    X_train_reshaped, X_test_reshaped = X_reshaped[train_index], X_reshaped[test_index] # LSTM exclusive
    y_train, y_test = y[train_index], y[test_index]

    # Random Forest Classifier training and prediction
    rf = RandomForestClassifier(n_estimators=100, random_state=42)
    rf.fit(X_train, y_train)
    y_pred_rf = rf.predict(X_test)
    y_pred_rf_proba = rf.predict_proba(X_test)[:, 1]

    # SVM Classifier training and prediction
    svm = SVC(kernel='rbf', class_weight='balanced', probability=True, random_state=42)
    svm.fit(X_train, y_train)
    y_pred_svm = svm.predict(X_test)
    y_pred_svm_proba = svm.predict_proba(X_test)[:, 1]

    # LSTM Classifier training and prediction
    lstm_model = create_lstm_model((X_train_reshaped.shape[1], X_train_reshaped.shape[2]))
    lstm_model.fit(X_train_reshaped, y_train, epochs=10, batch_size=64, verbose=0)
    y_pred_lstm = (lstm_model.predict(X_test_reshaped) > 0.5).astype(int)
    y_pred_lstm_proba = lstm_model.predict(X_test_reshaped).flatten()
```

Figure 4: Training and prediction for the three algorithms

```
# Jotting down all the important formulas needed for the output calculations for Random Forest
Accuracy = (TP+TN)/(TP+TN+FP+FN)
TPR = TP/(TP+FN)  # Sensitivity
TNR = TN/(TN+FP)  # Specificity
Precision = TP/(TP+FP)
F1_Score = 2*(Precision*TPR)/(Precision+TPR)
Error_Rate = (FP+FN)/(TP+TN+FP+FN)
BACC = (TPR+TNR)/2
FPR = FP/(FP+TN)
FNR = FN/(TP+FN)
TSS = TPR-FPR
BS = brier_score_loss(y_test, y_pred_rf_proba)
BSS = 1-BS/np.var(y_test)
AUC = roc_auc_score(y_test, y_pred_rf_proba)
HSS = 2*(TP*TN-FP*FN)/((TP+FN)*(FN+TN)+(TP+FP)*(FP+TN))
```

Figure 5: Random Forest Classifer's Metric Formulas

Random Forest Classifier Metrics:

	TP	TN	FP	FN	Accuracy	Sensitivity (TPR)	\
Fold 1	13.0	391.0	9.0	40.0	0.891832	0.245283	
Fold 2	1.0	399.0	1.0	51.0	0.884956	0.019231	
Fold 3	0.0	399.0	1.0	52.0	0.882743	0.000000	
Fold 4	10.0	388.0	12.0	42.0	0.880531	0.192308	
Fold 5	5.0	394.0	6.0	47.0	0.882743	0.096154	
Fold 6	11.0	395.0	5.0	41.0	0.898230	0.211538	
Fold 7	13.0	394.0	6.0	39.0	0.900442	0.250000	
Fold 8	5.0	395.0	5.0	47.0	0.884956	0.096154	
Fold 9	6.0	389.0	11.0	46.0	0.873894	0.115385	
Fold 10	7.0	394.0	6.0	45.0	0.887168	0.134615	
Average	7.1	393.8	6.2	45.0	0.886750	0.136067	

	Specificity (TNR)	Precision	F1 Score	Error Rate	\
Fold 1	0.9775	0.590909	0.346667	0.108168	
Fold 2	0.9975	0.500000	0.037037	0.115044	
Fold 3	0.9975	0.000000	0.000000	0.117257	
Fold 4	0.9700	0.454545	0.270270	0.119469	
Fold 5	0.9850	0.454545	0.158730	0.117257	
Fold 6	0.9875	0.687500	0.323529	0.101770	
Fold 7	0.9850	0.684211	0.366197	0.099558	
Fold 8	0.9875	0.500000	0.161290	0.115044	
Fold 9	0.9725	0.352941	0.173913	0.126106	
Fold 10	0.9850	0.538462	0.215385	0.112832	
Average	0.9845	0.476311	0.205302	0.113250	

	Balanced Accuracy (BACC)	FPR	FNR	Brier Score (BS)	\
Fold 1	0.611392	0.0225	0.754717	0.079431	
Fold 2	0.508365	0.0025	0.980769	0.081995	
Fold 3	0.498750	0.0025	1.000000	0.080470	
Fold 4	0.581154	0.0300	0.807692	0.076260	
Fold 5	0.540577	0.0150	0.903846	0.070012	
Fold 6	0.599519	0.0125	0.788462	0.073901	
Fold 7	0.617500	0.0150	0.750000	0.066950	
Fold 8	0.541827	0.0125	0.903846	0.081161	
Fold 9	0.543942	0.0275	0.884615	0.082223	
Fold 10	0.559808	0.0150	0.865385	0.072672	
Average	0.560283	0.0155	0.863933	0.076507	

	Brier Skill Score (BSS)	AUC	HSS	TSS
Fold 1	0.231132	0.840943	0.298518	0.222783
Fold 2	0.194624	0.836442	0.028760	0.016731
Fold 3	0.209600	0.856490	-0.004360	-0.002500
Fold 4	0.250951	0.891971	0.216688	0.162308
Fold 5	0.312322	0.903125	0.123518	0.081154
Fold 6	0.274122	0.873990	0.284810	0.199038
Fold 7	0.342394	0.904135	0.324612	0.235000
Fold 8	0.202813	0.858894	0.128965	0.083654
Fold 9	0.192380	0.836779	0.124269	0.087885
Fold 10	0.286192	0.900337	0.177537	0.119615
Average	0.249653	0.870311	0.170332	0.120567

Average Values Among All Folds:

TP	7.100000
TN	393.800000
FP	6.200000
FN	45.000000
Accuracy	0.886750
Sensitivity (TPR)	0.136067
Specificity (TNR)	0.984500
Precision	0.476311
F1 Score	0.205302
Error Rate	0.113250
Balanced Accuracy (BACC)	0.560283
FPR	0.015500
FNR	0.863933
Brier Score (BS)	0.076507
Brier Skill Score (BSS)	0.249653
AUC	0.870311
HSS	0.170332
TSS	0.120567

dtype: float64

Figure 6: Random Forest Classifer’s Metrics with Averages

SVM Metrics:

	TP	TN	FP	FN	Accuracy	Sensitivity (TPR)
Fold 1	13.0	391.0	9.0	40.0	0.891832	0.245283
Fold 2	1.0	399.0	1.0	51.0	0.884956	0.019231
Fold 3	0.0	399.0	1.0	52.0	0.882743	0.000000
Fold 4	10.0	388.0	12.0	42.0	0.880531	0.192308
Fold 5	5.0	394.0	6.0	47.0	0.882743	0.096154
Fold 6	11.0	395.0	5.0	41.0	0.898230	0.211538
Fold 7	13.0	394.0	6.0	39.0	0.900442	0.250000
Fold 8	5.0	395.0	5.0	47.0	0.884956	0.096154
Fold 9	6.0	389.0	11.0	46.0	0.873894	0.115385
Fold 10	7.0	394.0	6.0	45.0	0.887168	0.134615
Average	7.1	393.8	6.2	45.0	0.886750	0.136067

	Specificity (TNR)	Precision	F1 Score	Error Rate
Fold 1	0.9775	0.590909	0.346667	0.108168
Fold 2	0.9975	0.500000	0.037037	0.115044
Fold 3	0.9975	0.000000	0.000000	0.117257
Fold 4	0.9700	0.454545	0.270270	0.119469
Fold 5	0.9850	0.454545	0.158730	0.117257
Fold 6	0.9875	0.687500	0.323529	0.101770
Fold 7	0.9850	0.684211	0.366197	0.099558
Fold 8	0.9875	0.500000	0.161290	0.115044
Fold 9	0.9725	0.352941	0.173913	0.126106
Fold 10	0.9850	0.538462	0.215385	0.112832
Average	0.9845	0.476311	0.205302	0.113250

	Balanced Accuracy (BACC)	FPR	FNR	Brier Score (BS)
Fold 1	0.611392	0.0225	0.754717	0.079431
Fold 2	0.508365	0.0025	0.980769	0.081995
Fold 3	0.498750	0.0025	1.000000	0.080470
Fold 4	0.581154	0.0300	0.807692	0.076260
Fold 5	0.540577	0.0150	0.903846	0.070012
Fold 6	0.599519	0.0125	0.788462	0.073901
Fold 7	0.617500	0.0150	0.750000	0.066950
Fold 8	0.541827	0.0125	0.903846	0.081161
Fold 9	0.543942	0.0275	0.884615	0.082223
Fold 10	0.559808	0.0150	0.865385	0.072672
Average	0.560283	0.0155	0.863933	0.076507

	Brier Skill Score (BSS)	AUC	HSS	TSS
Fold 1	0.231132	0.840943	0.298518	0.222783
Fold 2	0.194624	0.836442	0.028760	0.016731
Fold 3	0.209600	0.856490	-0.004360	-0.002500
Fold 4	0.250951	0.891971	0.216688	0.162308
Fold 5	0.312322	0.903125	0.123518	0.081154
Fold 6	0.274122	0.873990	0.284810	0.199038
Fold 7	0.342394	0.904135	0.324612	0.235000
Fold 8	0.202813	0.858894	0.128965	0.083654
Fold 9	0.192380	0.836779	0.124269	0.087885
Fold 10	0.286192	0.900337	0.177537	0.119615
Average	0.249653	0.870311	0.170332	0.120567

Average Values Among All Folds:

TP	7.100000
TN	393.800000
FP	6.200000
FN	45.000000
Accuracy	0.886750
Sensitivity (TPR)	0.136067
Specificity (TNR)	0.984500
Precision	0.476311
F1 Score	0.205302
Error Rate	0.113250
Balanced Accuracy (BACC)	0.560283
FPR	0.015500
FNR	0.863933
Brier Score (BS)	0.076507
Brier Skill Score (BSS)	0.249653
AUC	0.870311
HSS	0.170332
TSS	0.120567

dtype: float64

Figure 7: SVM's Metrics with Averages

LSTM Metrics:

	TP	TN	FP	FN	Accuracy	Sensitivity (TPR)	\
Fold 1	13.0	391.0	9.0	40.0	0.891832	0.245283	
Fold 2	1.0	399.0	1.0	51.0	0.884956	0.019231	
Fold 3	0.0	399.0	1.0	52.0	0.882743	0.000000	
Fold 4	10.0	388.0	12.0	42.0	0.880531	0.192308	
Fold 5	5.0	394.0	6.0	47.0	0.882743	0.096154	
Fold 6	11.0	395.0	5.0	41.0	0.898230	0.211538	
Fold 7	13.0	394.0	6.0	39.0	0.900442	0.250000	
Fold 8	5.0	395.0	5.0	47.0	0.884956	0.096154	
Fold 9	6.0	389.0	11.0	46.0	0.873894	0.115385	
Fold 10	7.0	394.0	6.0	45.0	0.887168	0.134615	
Average	7.1	393.8	6.2	45.0	0.886750	0.136067	

	Specificity (TNR)	Precision	F1 Score	Error Rate	\
Fold 1	0.9775	0.590909	0.346667	0.108168	
Fold 2	0.9975	0.500000	0.037037	0.115044	
Fold 3	0.9975	0.000000	0.000000	0.117257	
Fold 4	0.9700	0.454545	0.270270	0.119469	
Fold 5	0.9850	0.454545	0.158730	0.117257	
Fold 6	0.9875	0.687500	0.323529	0.101770	
Fold 7	0.9850	0.684211	0.366197	0.099558	
Fold 8	0.9875	0.500000	0.161290	0.115044	
Fold 9	0.9725	0.352941	0.173913	0.126106	
Fold 10	0.9850	0.538462	0.215385	0.112832	
Average	0.9845	0.476311	0.205302	0.113250	

	Balanced Accuracy (BACC)	FPR	FNR	Brier Score (BS)	\
Fold 1	0.611392	0.0225	0.754717	0.079431	
Fold 2	0.508365	0.0025	0.980769	0.081995	
Fold 3	0.498750	0.0025	1.000000	0.080470	
Fold 4	0.581154	0.0300	0.807692	0.076260	
Fold 5	0.540577	0.0150	0.903846	0.070012	
Fold 6	0.599519	0.0125	0.788462	0.073901	
Fold 7	0.617500	0.0150	0.750000	0.066950	
Fold 8	0.541827	0.0125	0.903846	0.081161	
Fold 9	0.543942	0.0275	0.884615	0.082223	
Fold 10	0.559808	0.0150	0.865385	0.072672	
Average	0.560283	0.0155	0.863933	0.076507	

	Brier Skill Score (BSS)	AUC	HSS	TSS
Fold 1	0.231132	0.840943	0.298518	0.222783
Fold 2	0.194624	0.836442	0.028760	0.016731
Fold 3	0.209600	0.856490	-0.004360	-0.002500
Fold 4	0.250951	0.891971	0.216688	0.162308
Fold 5	0.312322	0.903125	0.123518	0.081154
Fold 6	0.274122	0.873990	0.284810	0.199038
Fold 7	0.342394	0.904135	0.324612	0.235000
Fold 8	0.202813	0.858894	0.128965	0.083654
Fold 9	0.192380	0.836779	0.124269	0.087885
Fold 10	0.286192	0.900337	0.177537	0.119615
Average	0.249653	0.870311	0.170332	0.120567

Average Values Among All Folds:

TP	7.100000
TN	393.800000
FP	6.200000
FN	45.000000
Accuracy	0.886750
Sensitivity (TPR)	0.136067
Specificity (TNR)	0.984500
Precision	0.476311
F1 Score	0.205302
Error Rate	0.113250
Balanced Accuracy (BACC)	0.560283
FPR	0.015500
FNR	0.863933
Brier Score (BS)	0.076507
Brier Skill Score (BSS)	0.249653
AUC	0.870311
HSS	0.170332
TSS	0.120567

dtype: float64

Figure 8: LSTM's Metrics with Averages

```
Time taken to output: 72.19 seconds
```

Figure 9: Total time it took to run the code in seconds

Github link: https://github.com/vaibhavbora11/Data-Mining-Final-Project

Dataset Source: https://archive.ics.uci.edu/dataset/222/bank+marketing