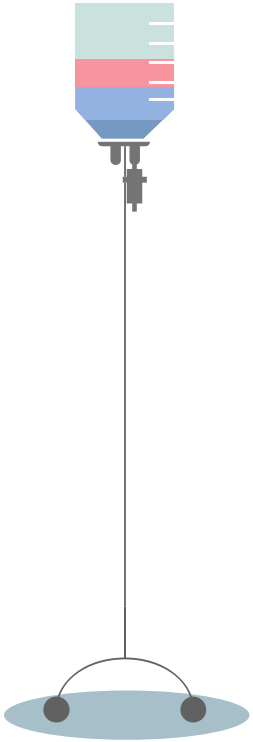
Prediction of Stroke

Capstone Presentation- 10 Apr 2021

By S Buvana

TABLE OF CONTENTS



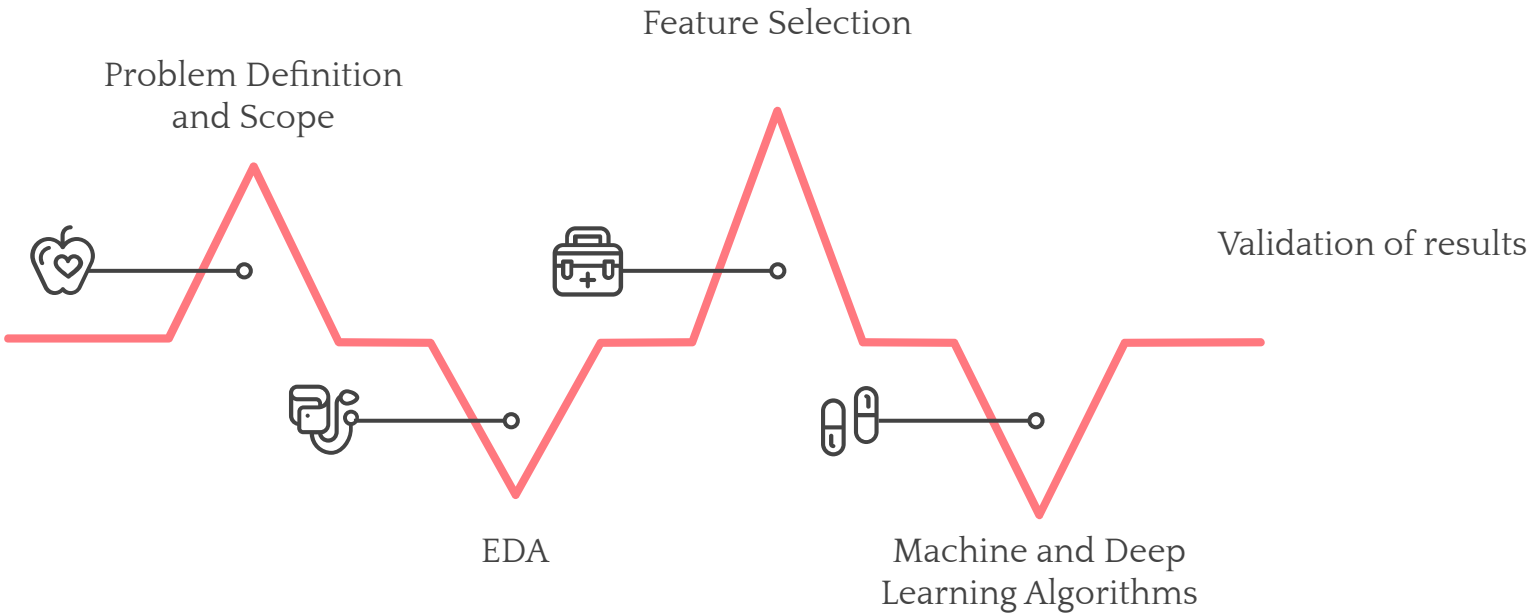
Problem Statement

To predict Stroke occurrences

Scope

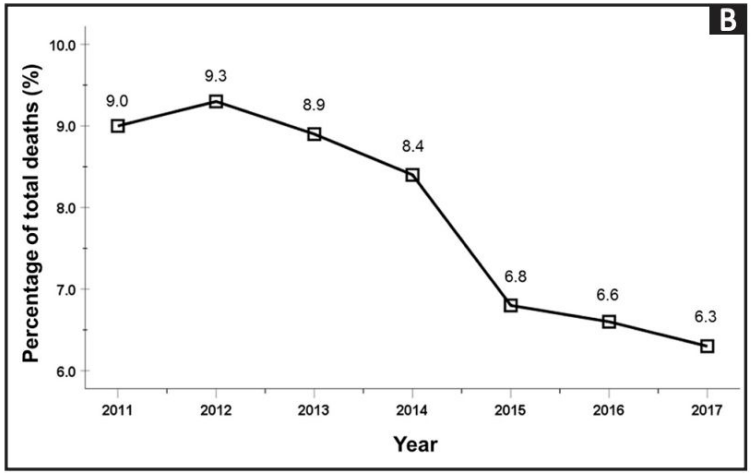
To build Machine and Deep Learning models on identified features. Use of Accuracy performance metrics on datasets to evaluate robustness of models.

Data Process



Background

- Very much of a concern in Singapore
- Mortality rates from Stroke -9%to 6.3% from 2011 to 2017.
- Fourth leading cause of death in Singapore
- Need for improvements
- Deploying prediction models to study the trends will help in early detection of Stroke



Source: Annals Academy of Medicine, Singapore

Outcomes

Desirable Business Outcome

To detect Stroke accurately and quickly which brings about cost savings.

Desirable Data Outcome

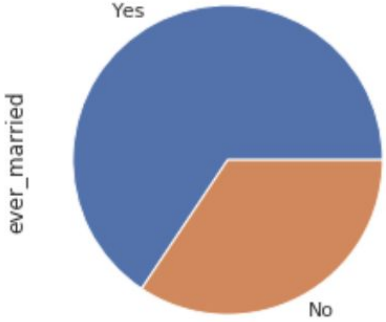
The model to be deployed to detect Stroke with speed and accuracy.

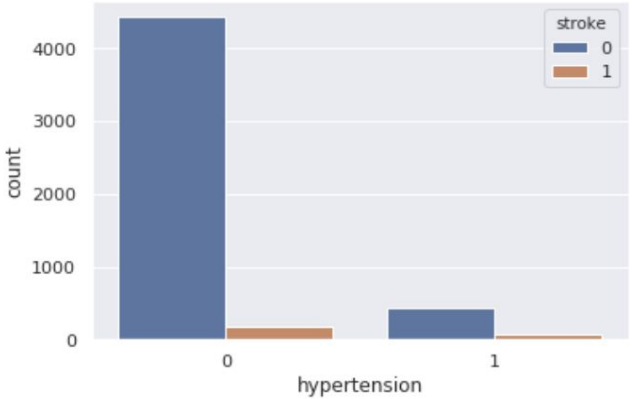
Data

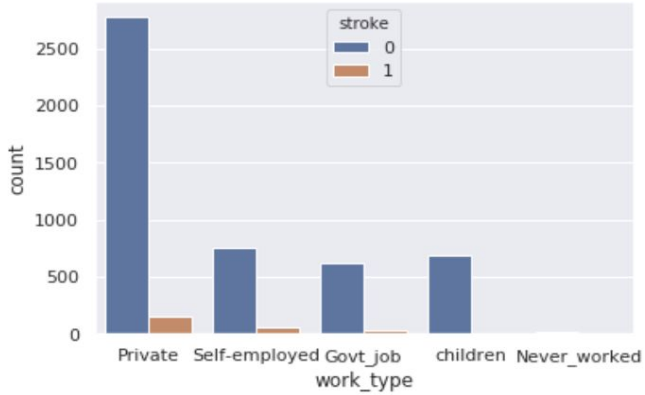
- Kaggle dataset- Stroke prediction
- 5110 Rows & 12 Columns- Target variable is **Stroke**
- 5 Categorical variables (Dummy Encoding)
- Normalisation of dataset

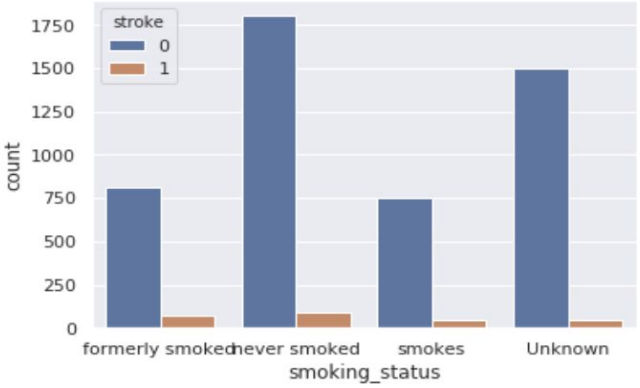
	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
5	56669	Male	81.0	0	0	Yes	Private	Urban	186.21	29.0	formerly smoked	1
6	53882	Male	74.0	1	1	Yes	Private	Rural	70.09	27.4	never smoked	1
7	10434	Female	69.0	0	0	No	Private	Urban	94.39	22.8	never smoked	1
8	27419	Female	59.0	0	0	Yes	Private	Rural	76.15	NaN	Unknown	1
9	60491	Female	78.0	0	0	Yes	Private	Urban	58.57	24.2	Unknown	1

EDA


Yes
No
ever_married

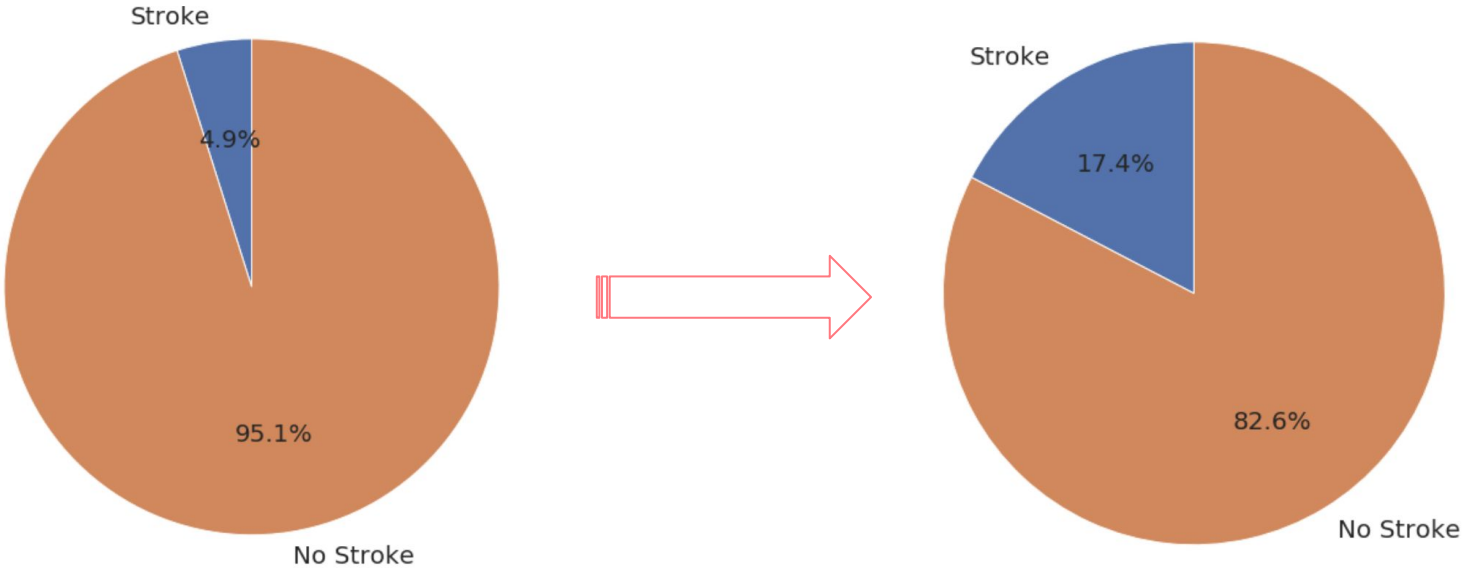

stroke
0
1
4000
3000
2000
1000
0
count
0
1
hypertension


stroke
0
1
2500
2000
1500
1000
500
0
count
Private
Self-employed
Govt_job
children
Never_worked
work_type


stroke
0
1
1750
1500
1250
1000
750
500
250
0
count
formerly smoked
never smoked
smokes
Unknown
smoking_status

EDA

- Unbalanced dataset with only 4.9% of data points are Stroke patients
- Resampling method to artificially inflate the minority class (with Stroke)
- From 4.9% to around 17.4%- Model analysis were run on resampled dataset



Feature Selection

Forward Feature Selection(FFS)

Predictor Columns:'age', 'work_type_4', 'avg_glucose_level', 'ever_married_0', 'heart_disease', 'hypertension', 'bmi', 'work_type_2', 'smoking_status_2', 'gender_1'

PCA

15 PCA components

Machine Learning

- Machine learning algorithm run on both datasets with variables identified by feature selection methods.
- Train/Test split of 80%/20%
- Cross validation
- Hyper parameter tuning by GridSearch CV

Logistic Regression

KNN Classification

Support Vector

Random Forest

Ada Boosting

Deep Learning

- Deep learning algorithm run on both datasets with variables identified by feature selection methods.
- Train/Test split of 80%/20%
- Cross validation
- Build **Artificial Neural Network**

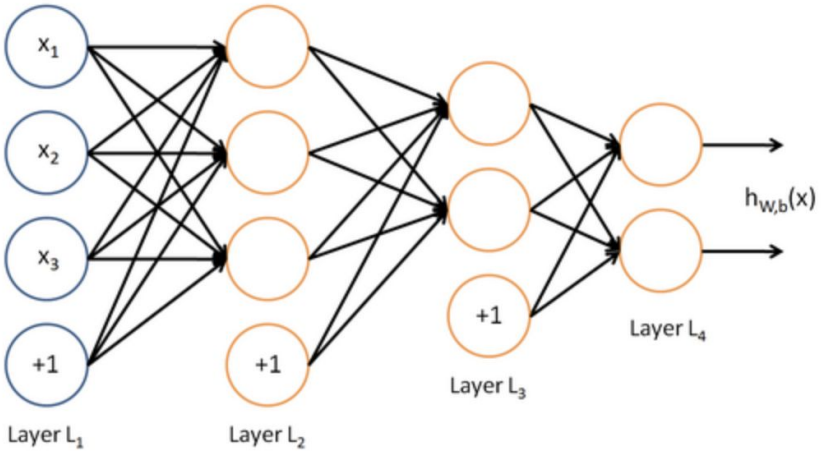


Figure 1 — Representation of a neural network

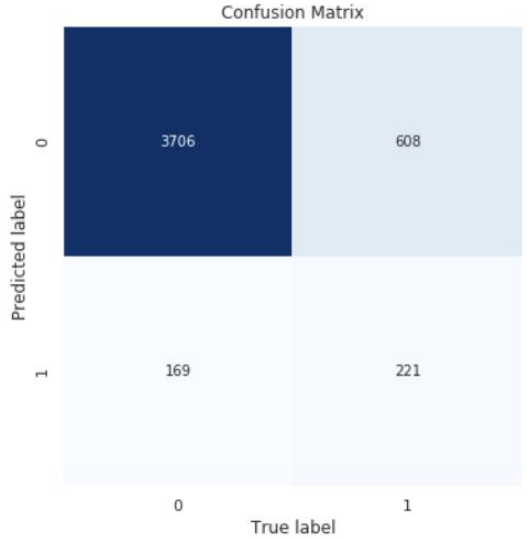
Machine Learning

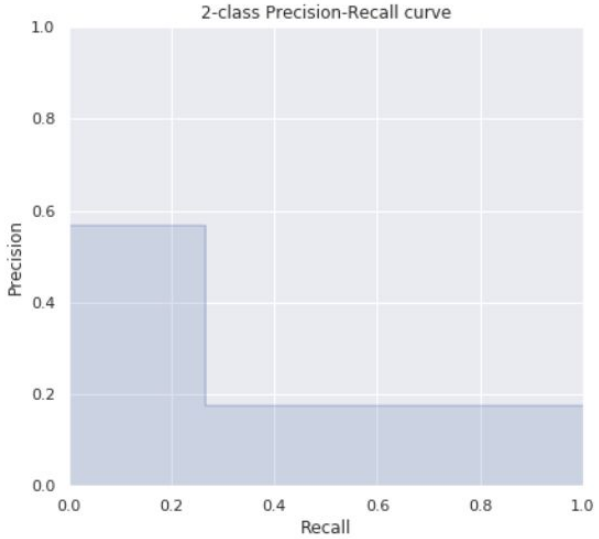
On predictors from Forward Feature Selection

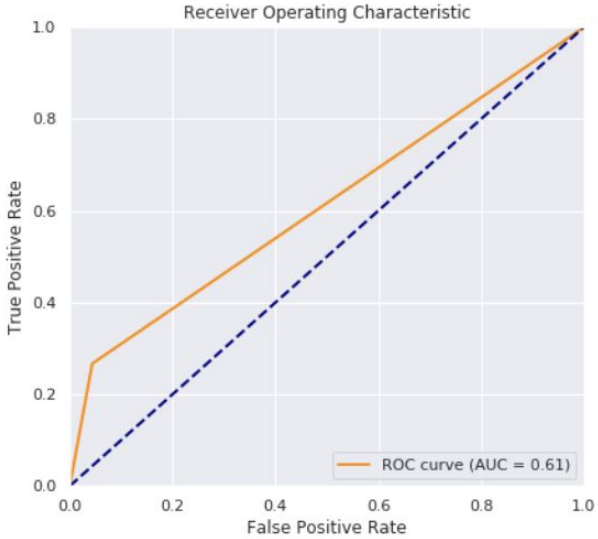
Logistic Regression-FFS

Training Set

```
*************
* Logistic *
*************
Accuracy : 0.8348 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.5667 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.2666 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.6115                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



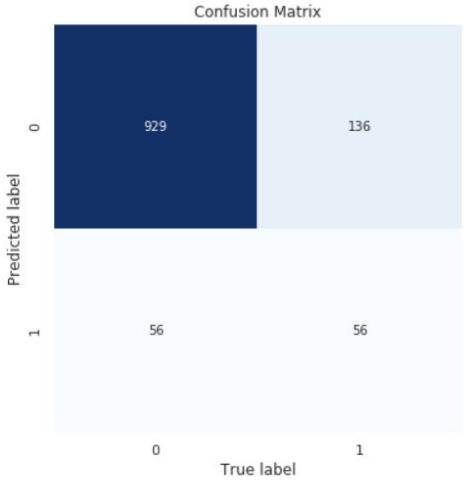


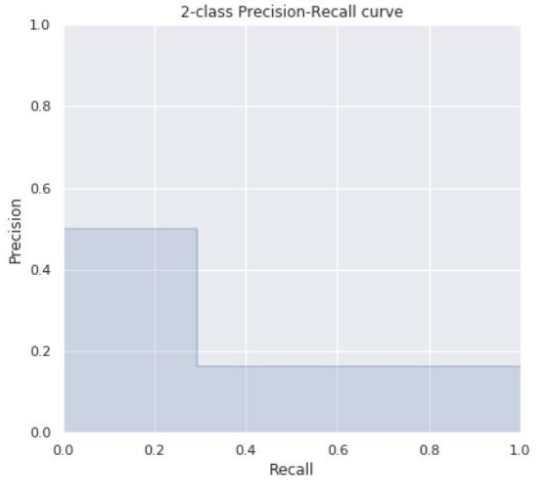


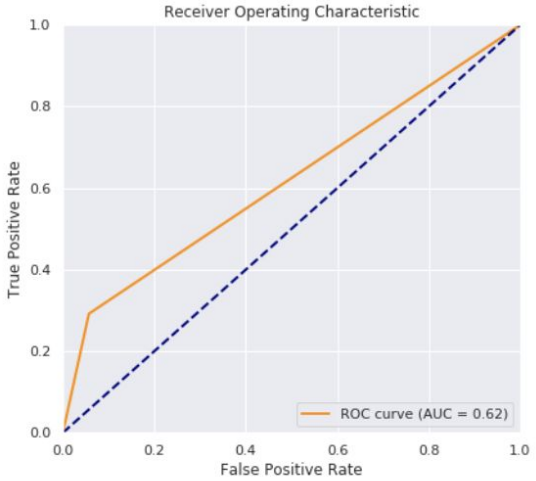
Logistic Regression-FFS

Test Set

```
Accuracy : 0.8369 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.5000 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.2917 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.6174                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



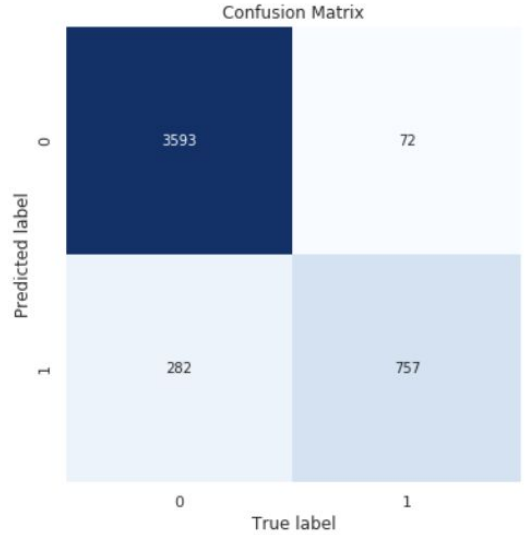


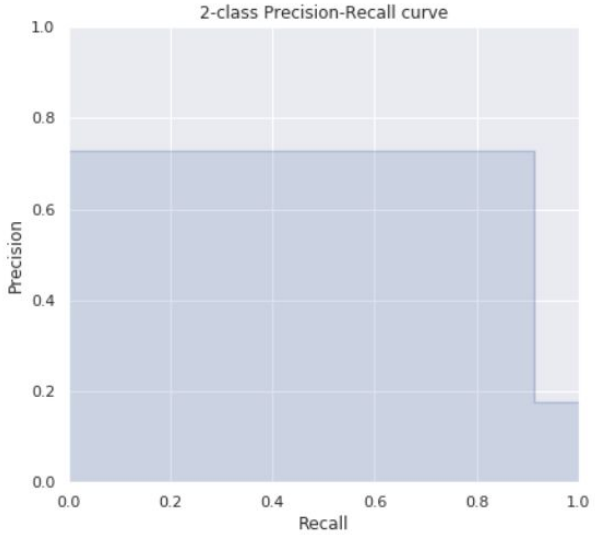


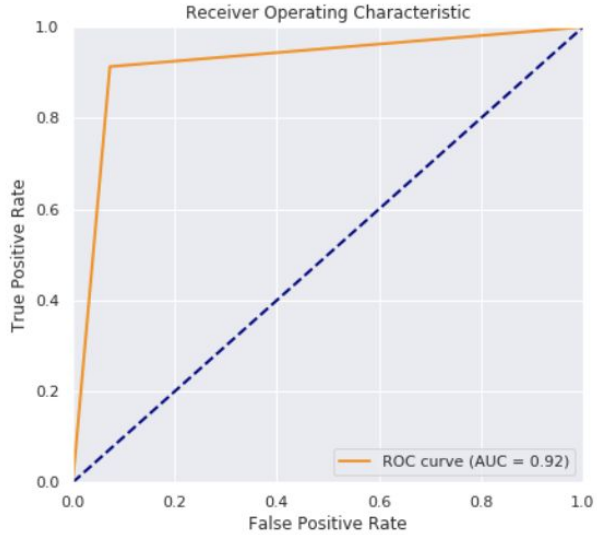
KNN-FFS

Training Set

```
********************
* KNN Classifier *
********************
Accuracy : 0.9247 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.7286 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.9131 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.9202                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



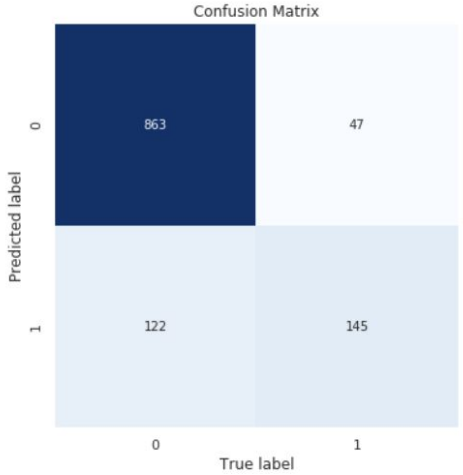


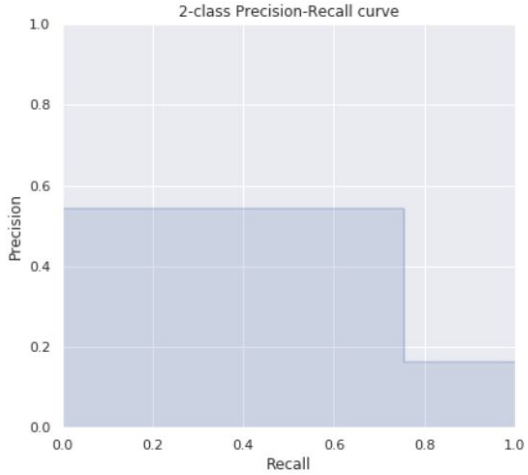


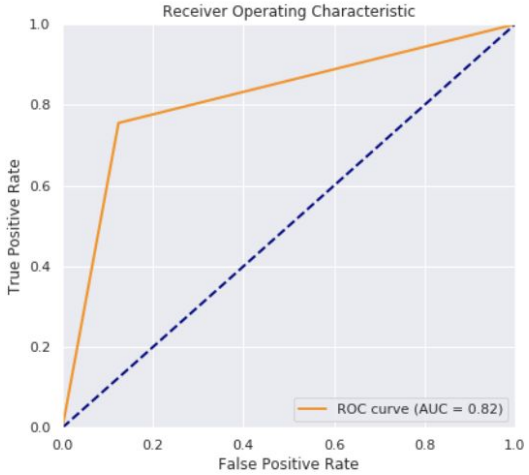
KNN-FFS

Test Set

```
Accuracy : 0.8564 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.5431 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.7552 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.8157                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



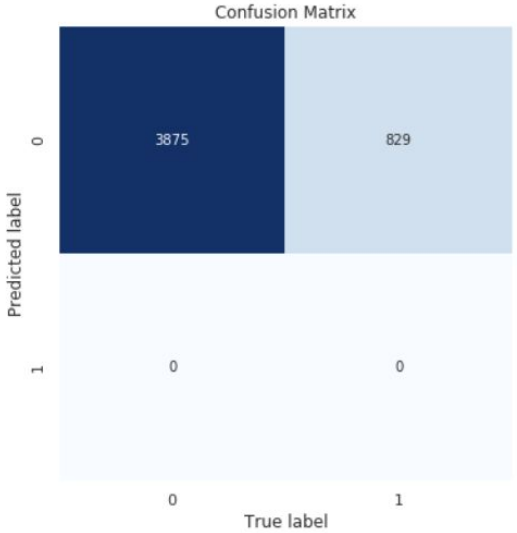


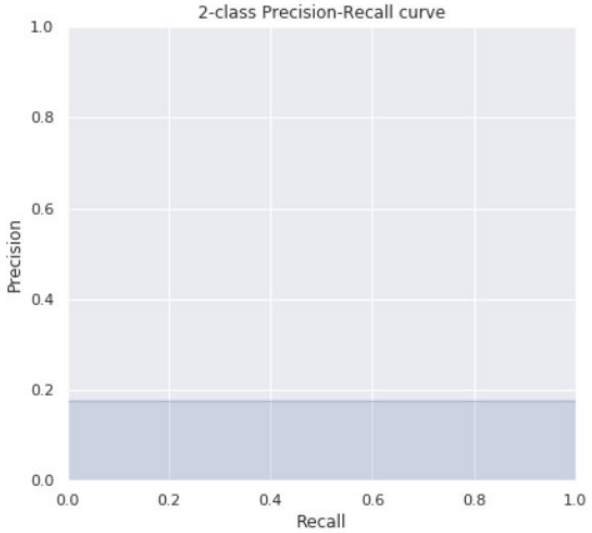


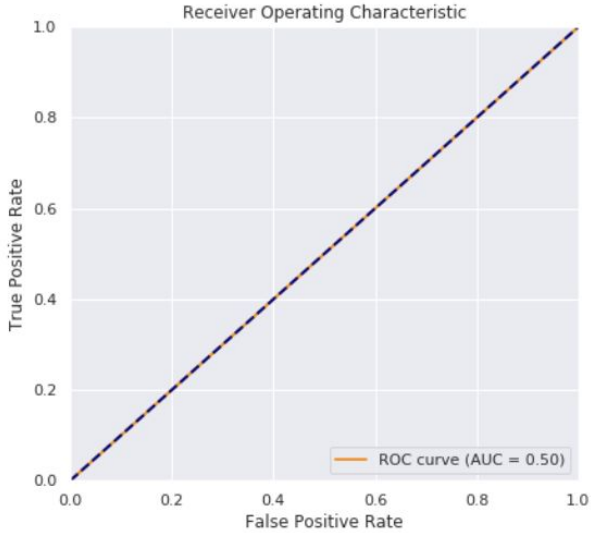
SVM-FFS

Training Set

```
********
* SVM *
********
Accuracy : 0.8238 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.0000 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.0000 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.5000                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



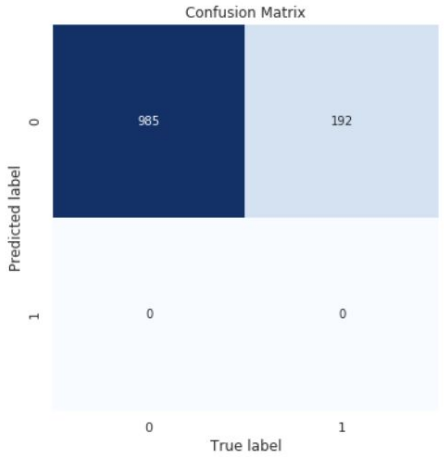


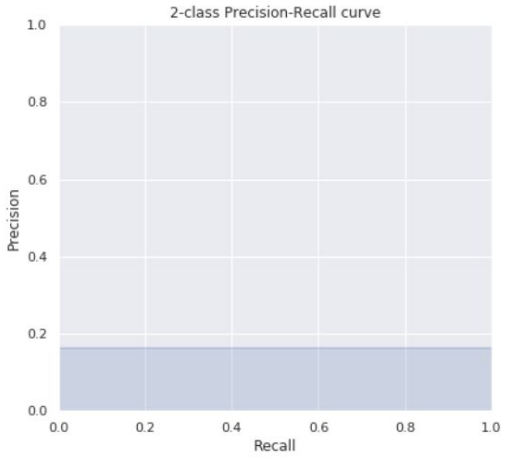


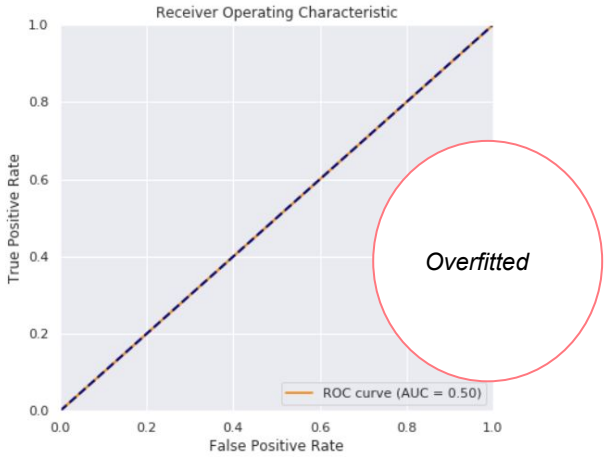
SVM-FFS

Test Set

```
Accuracy : 0.8369 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.0000 [TP / (TP + FP)] Not to label a negative sample as positive.       Best: 1, Worst: 0
Recall   : 0.0000 [TP / (TP + FN)] Find all the positive samples.                    Best: 1, Worst: 0
ROC AUC  : 0.5000                                                                    Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



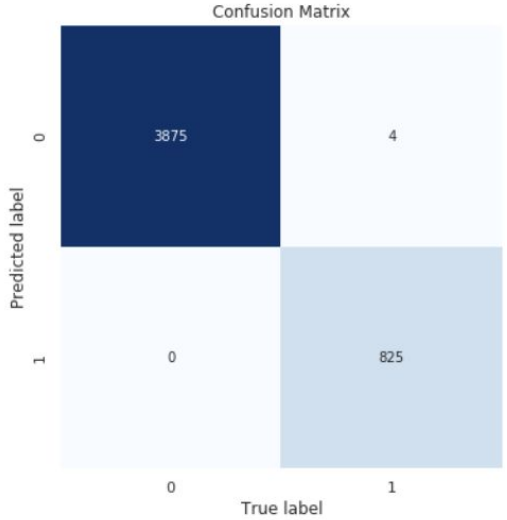


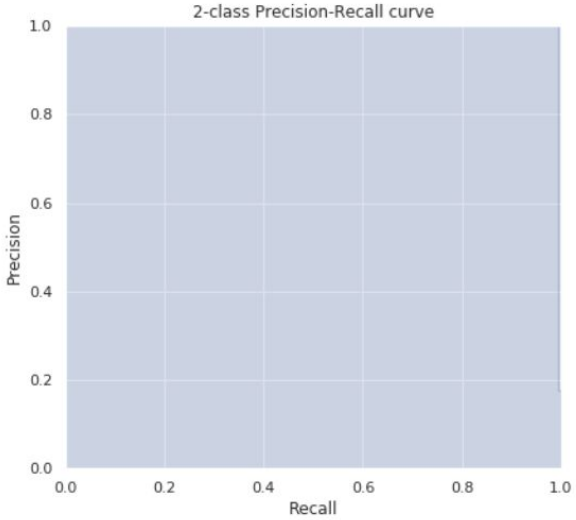


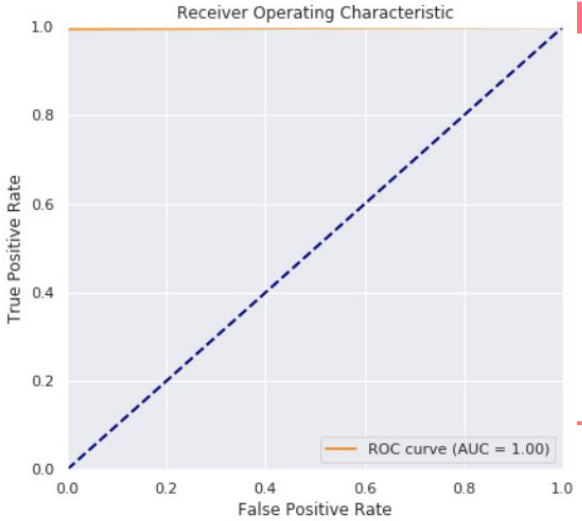
Random Forest-FFS

Training Set

```
******************
* Random Forest *
******************
Accuracy : 0.9991 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 1.0000 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.9952 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.9976                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



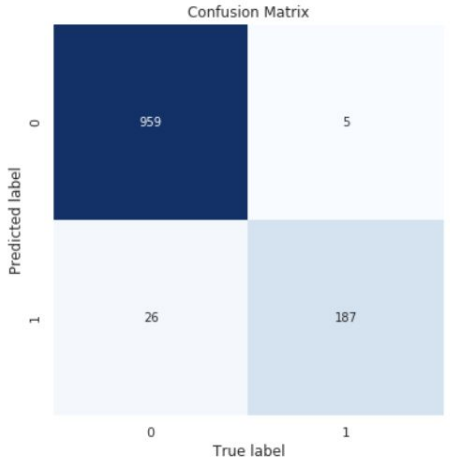


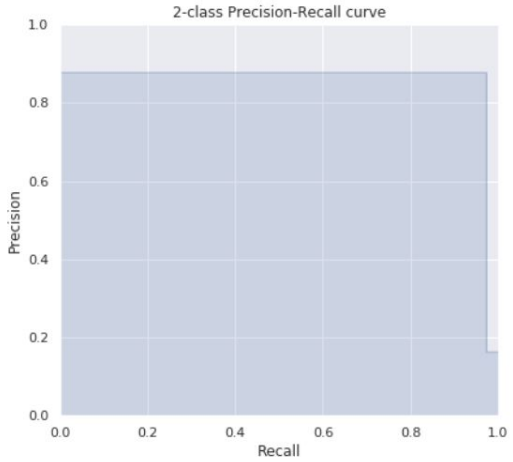


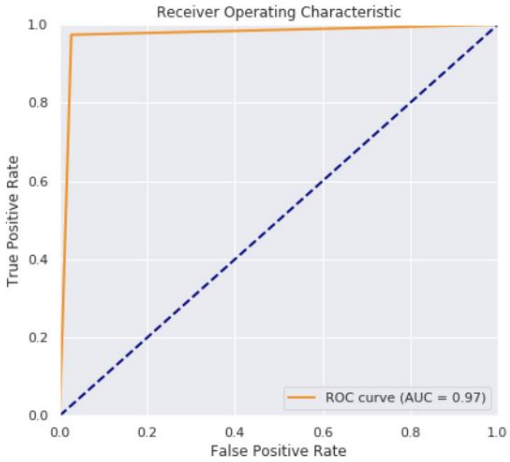
Random Forest-FFS

Test Set

```
Accuracy : 0.9737 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.8779 [TP / (TP + FP)] Not to label a negative sample as positive.       Best: 1, Worst: 0
Recall   : 0.9740 [TP / (TP + FN)] Find all the positive samples.                    Best: 1, Worst: 0
ROC AUC  : 0.9738                                                                    Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



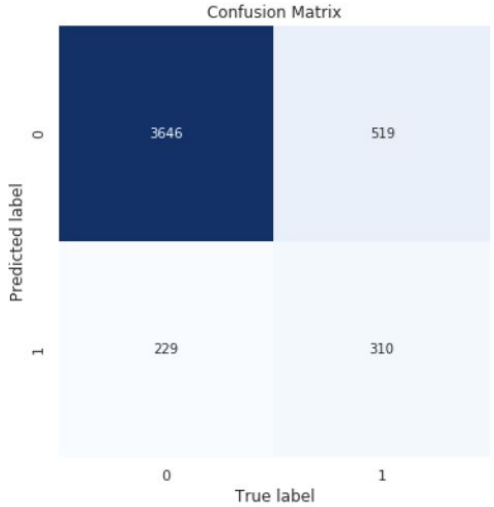


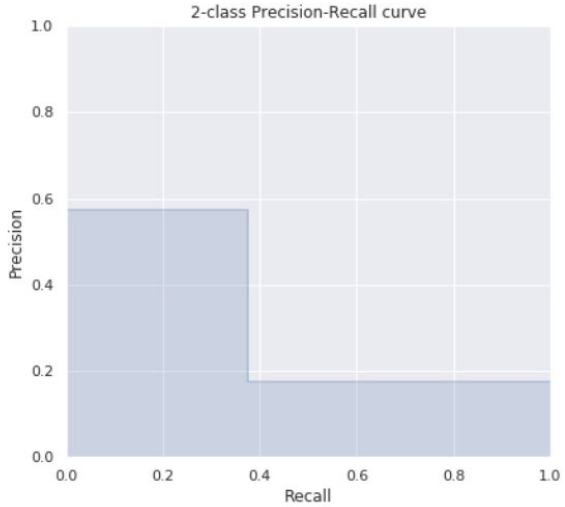


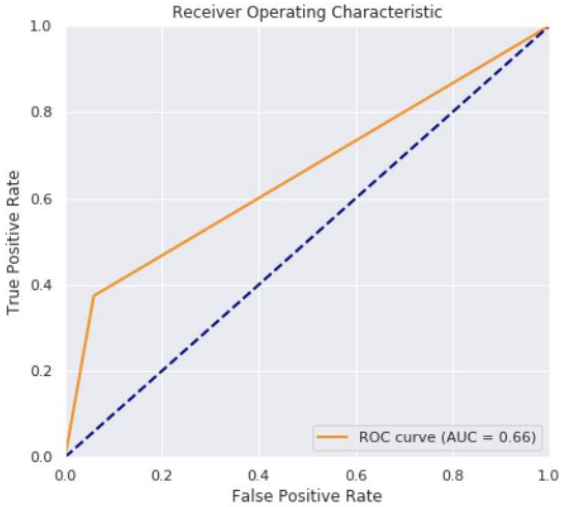
Boosting-FFS

Training Set

```
*************
* Boosting *
*************
Accuracy : 0.8410 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.5751 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.3739 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.6574                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```



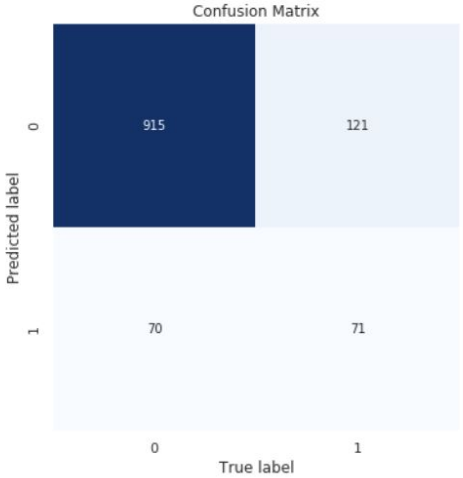


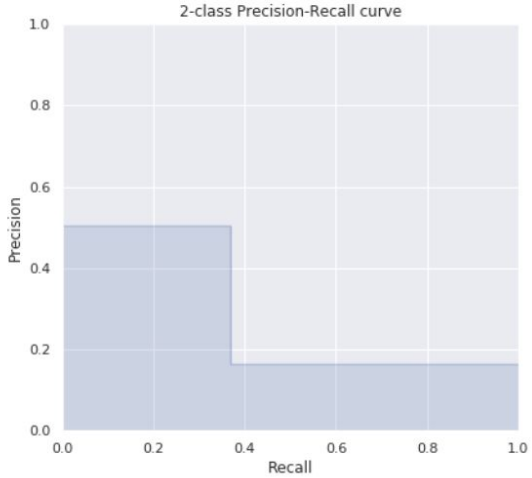


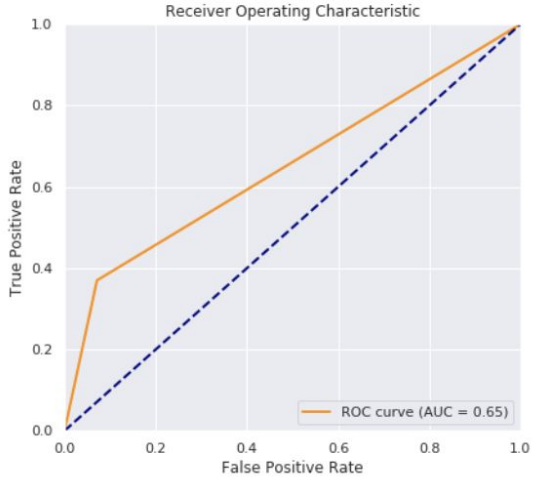
Boosting-FFS

Test Set

```
Accuracy : 0.8377 [TP / N] Proportion of predicted labels that match the true labels. Best: 1, Worst: 0
Precision: 0.5035 [TP / (TP + FP)] Not to label a negative sample as positive.        Best: 1, Worst: 0
Recall   : 0.3698 [TP / (TP + FN)] Find all the positive samples.                     Best: 1, Worst: 0
ROC AUC  : 0.6494                                                                     Best: 1, Worst: < 0.5
----------------------------------------------------------------------------------------------------------
TP: True Positives, FP: False Positives, TN: True Negatives, FN: False Negatives, N: Number of samples
```







Performance Metrics- FFS dataset

Train Dataset	Logistic Regression	KNN	SVM	Random Forest	Ada Boost
Forward Feature Selection -10 X columns	0.834	0.924	0.823	**0.999**	0.841
Forward Feature Selection -10 X columns with GridSearch CV	0.835	**1.0**	0.824	0.999	0.856

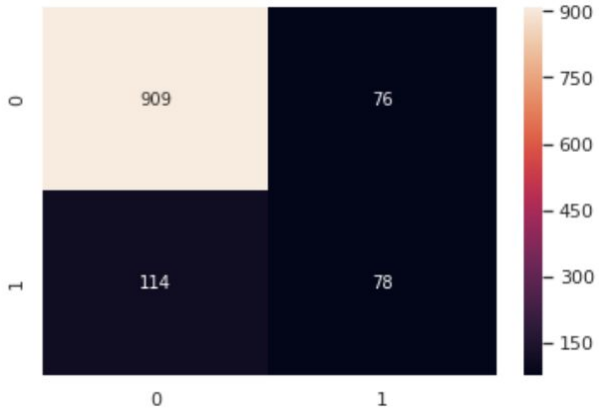
Deep Learning

On predictors from Forward Feature Selection

Neural Network- FFS

- Keras sequential model
- ANN model with 15 Neurons in L1 and 1 hidden layer
- Drop out rate :0.2
- Optimizer: Adam
- Loss: Binary cross entropy
- Metrics: Accuracy
- 100 epoch runs

Model Results



-Accuracy score of prediction model 84.8%
-Good classifier

Machine Learning

On predictors from PCA

Performance Metrics- PCA dataset

Train Dataset	Logistic Regression	KNN	SVM	Random Forest	Ada Boost
PCA-15 PCA components	0.835	0.924	0.824	**0.999**	0.862
PCA-15 PCA components with GridSearch CV	0.835	0.835	0.824	**1.00**	0.936

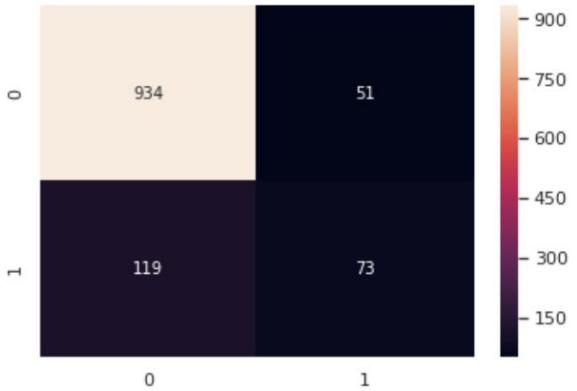
Deep Learning

On predictors from PCA

Neural Network- PCA

- Keras sequential model
- ANN model with 15 Neurons in L1 and 1 hidden layer
- Drop out rate :0.2
- Optimizer: Adam
- Loss: Binary cross entropy
- Metrics: Accuracy
- 100 epoch runs

Model Results

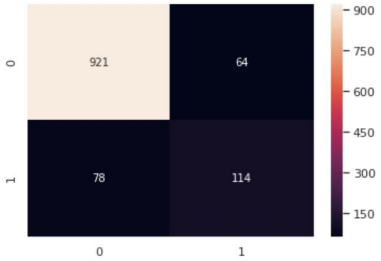


-Accuracy score of prediction model 85.6%
-Good classifier

Neural Network- PCA

Parameter tuning-
Change batch size to
10

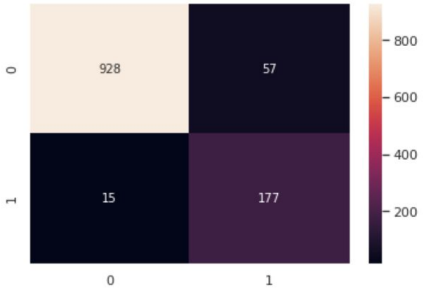
Model Results



-Accuracy score of prediction model 87.9%
-Slight improvement

Parameter tuning-
Change epoch size to
150

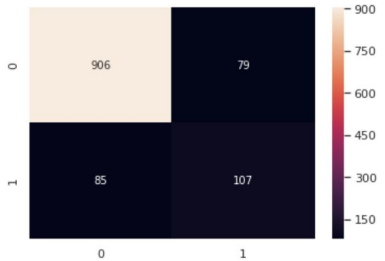
Model Results



-Accuracy score of prediction model 93.9%
-Slight improvement

Parameter tuning-
Change dropout rate
to 0.1

Model Results



-Accuracy score of prediction model 86%
-Slight improvement

Conclusions

Based on the data collected, we can use Machine Learning model — Random Forest to help with the prediction of Stroke occurrences. It provided the best accuracy at the rate of **99.9%** across both datasets.

AUC Score of **1.00** for rain and **0.97** for test indicates a good classifier and overfitting is unlikely.The recall and precision are high above 90% indicating the exactness and proper classification of data points, which is crucial in detecting diseases.

Medical institutes can detect Stroke in **accurate and timely manner** to reduce mortality rates further.

THANK YOU

RESOURCE LINKS

- https://www.annals.edu.sg/pdf/48VolNo10Oct2019/V48N10p310.pdf
- https://www.kaggle.com/kanishkkavdia/stroke-prediction-using-ann-deep-learning-model

CREDITS

This is where you give credit to the ones who are part of this project.
Did you like the resources on this template? Get them for free at our other websites.

- Presentation template by Slidesgo
- Icons by Flaticon
- Infographics by Freepik
- Images created by Freepik
- Text & Image slide photo created by Freepik.com

Appendices

Encoding and Correlation

```
bles to run models

pd.get_dummies(data = Stroke_df, columns = ['gender','
               prefix = ['gender','ever_married', 'work_
ad()
```

	age	hypertension	heart_disease	avg_glucose_level	bmi	stroke
0	67	0	1	228.69	36.600000	1
1	61	0	0	202.21	28.893237	1
2	80	0	1	105.92	32.500000	1
3	49	0	0	171.23	34.400000	1
4	79	1	0	174.12	24.000000	1

5 rows × 21 columns

```
corr_matrix = Stroke_df1.corr()
print(corr_matrix["stroke"].sort_values(ascending=False))
```

```
stroke                1.000000
age                   0.245109
heart_disease         0.134905
avg_glucose_level     0.131991
hypertension          0.127891
ever_married_1        0.108299
smoking_status_1      0.064683
work_type_3           0.062150
bmi                   0.038912
Residence_type_1      0.015415
work_type_2           0.011927
gender_1              0.009081
smoking_status_3      0.008920
work_type_0           0.002660
smoking_status_2     -0.004163
gender_0             -0.009081
work_type_1          -0.014885
Residence_type_0     -0.015415
smoking_status_0     -0.055924
work_type_4          -0.083888
ever_married_0       -0.108299
Name: stroke, dtype: float64
```

ANN

```
ann_model=keras.Sequential([
    keras.layers.Dense(15,input_shape=[10], activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(20,  activation='relu'),
    keras.layers.Dropout(0.2),
    keras.layers.Dense(1,activation="sigmoid")
])

print(ann_model.summary())
ann_model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])

ann_model.summary()
fitted_model =ann_model.fit(X_train, y_train, batch_size = 4, nb_epoch = 100)
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 15)                165
_________________________________________________________________
dropout (Dropout)            (None, 15)                0
_________________________________________________________________
dense_1 (Dense)              (None, 20)                320
_________________________________________________________________
dropout_1 (Dropout)          (None, 20)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 21
=================================================================
Total params: 506
Trainable params: 506
Non-trainable params: 0
```

Performance Metrics

If resampling is tuned to reflect 50%-50% of target variable

Performance Metrics- PCA dataset

Generally Models didn't perform as well in comparison

Train Dataset	Logistic Regression	KNN	SVM	Random Forest	Ada Boost
PCA-15 PCA components	0.776	0.934	0.778	**0.999**	0.811
PCA-15 PCA components with GridSearch CV	0.776	0.835	0.824	**0.999**	0.936