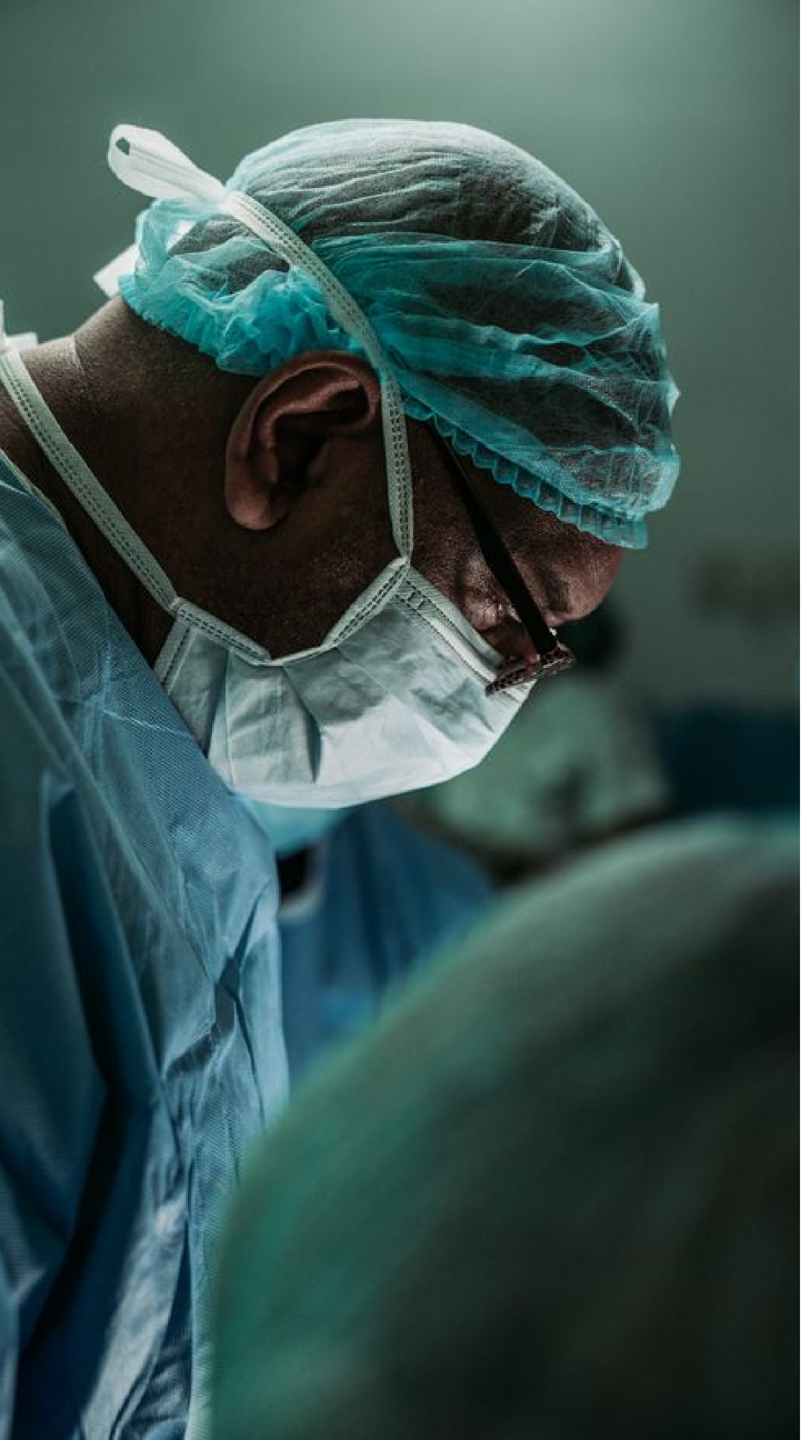


Stroke Prediction Algorithm

“This is about saving lives”

VU University Medical Center, Amsterdam
30/04/2019

Mathieu Stremsdoerfer

Dataset
https://www.kaggle.com/asaumya/healthcare-dataset-stroke-data
Program: Github

We were given three days to...

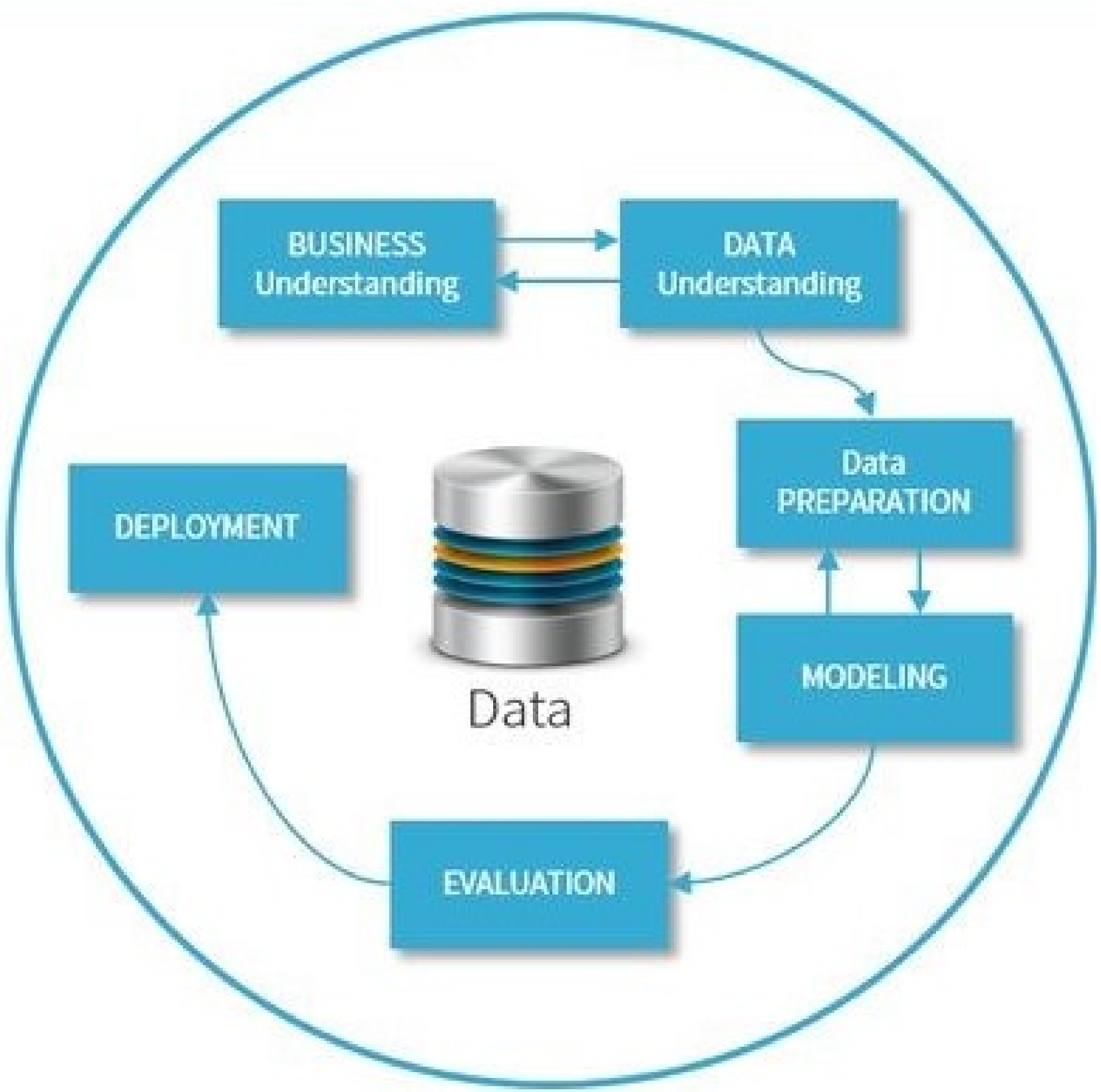
Build an algorithm to predict if patient will suffer from stroke or not

Try out a few different Learning Algorithms, evaluate them, compare them and choose the best model

Walk you through the solution and present the next steps

Crisp-DM was used*

*Cross Industry Standard Process for Data Mining



Stroke Detection Objectives, KPIs and metrics

Save lives

Correctly categorize patients at risk of stroke = **Maximize DETECTION**

Be efficient

Limit patients visits if not necessary = **Minimize FALSE ALARMS**

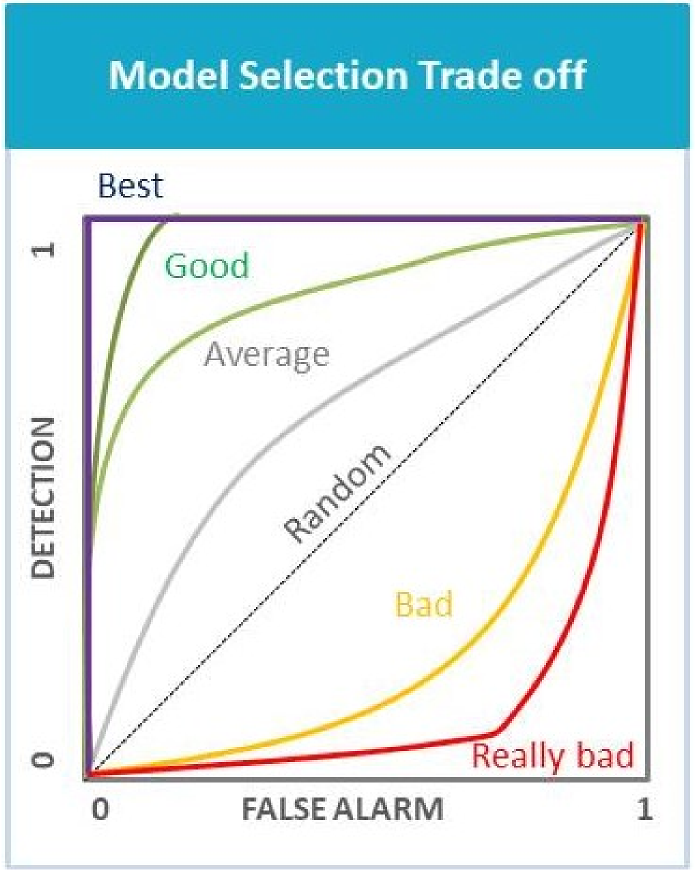
KPIs

Performance gain versus **logistic regression model**

>**90% Stroke detection**

Metrics

Minimize False Alarms when Detection is maximal



DETECTION RATE = SAVE LIVES / (SAVE LIVES+MISS) ; FALSE ALARM RATE = FALSE ALARM/(SAVE LIVES+FALSE ALARM)

Recommendation

Gradient Boosting tuned model: +19% higher performance versus Logistic regression

+5.7% versus non tuned

Prediction for 100 people		Predicted: no stroke	Predicted: stroke
reality	no stroke	93.5%	4.9%
	stroke	0.15%	1.5%

Detection rate: 91.3% vs 78%

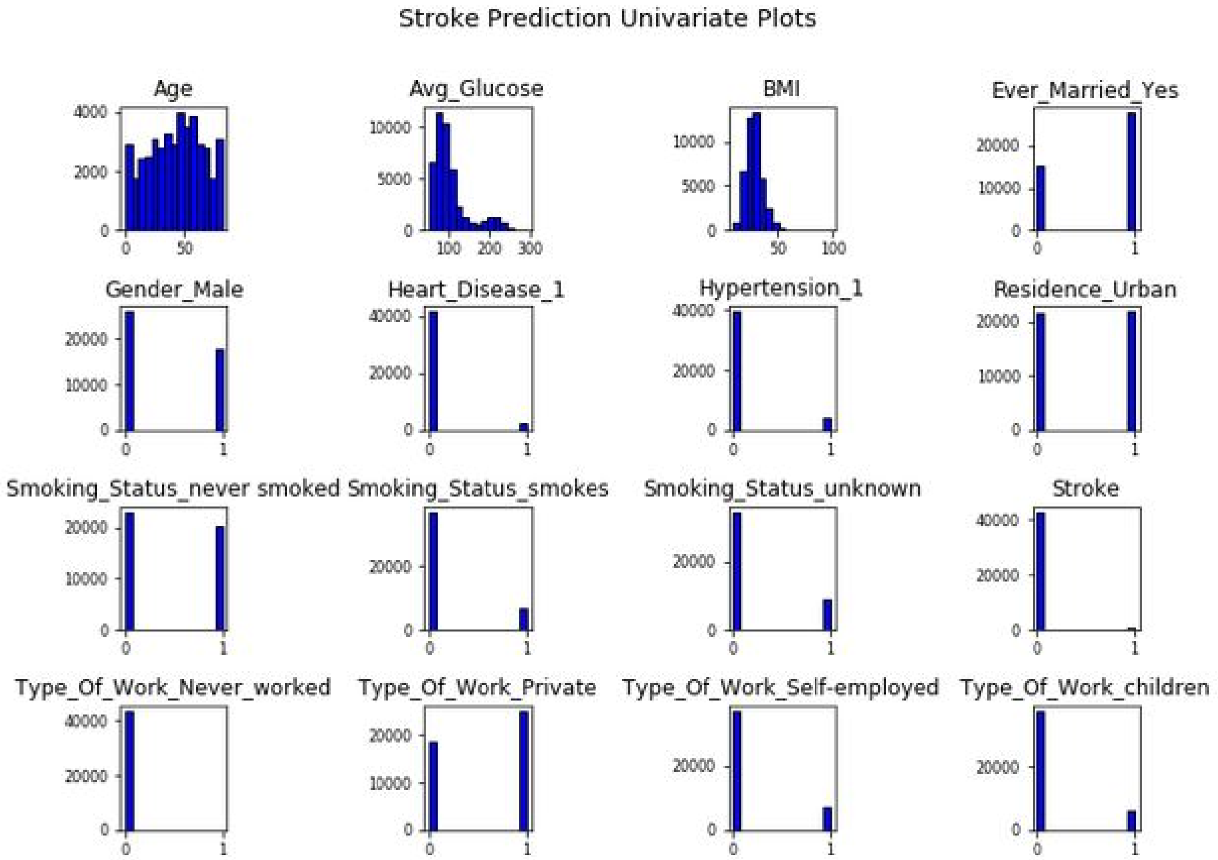
False alarm rate:

4.96% vs 25%

Stroke missed: 8.7% vs 18%

How we got there…

Learning from the data exploration



- **Imbalanced binary classification**

  43385 observations, 783 strokes (1.8%)

- **Mix data (3 Numeric and 13 Categorical)**

- **Missing data**

  30% smoking, 3% BMI

- **Irrelevant data**

  Children (0.03% stroke), Gender = Other

- **Incorrect data: IDs**

  > 18000 duplicates with non matching data

- **Some correlations, but low for all attributes except age**

Higher % stroke tend to increase with combination of following attributes

Older people

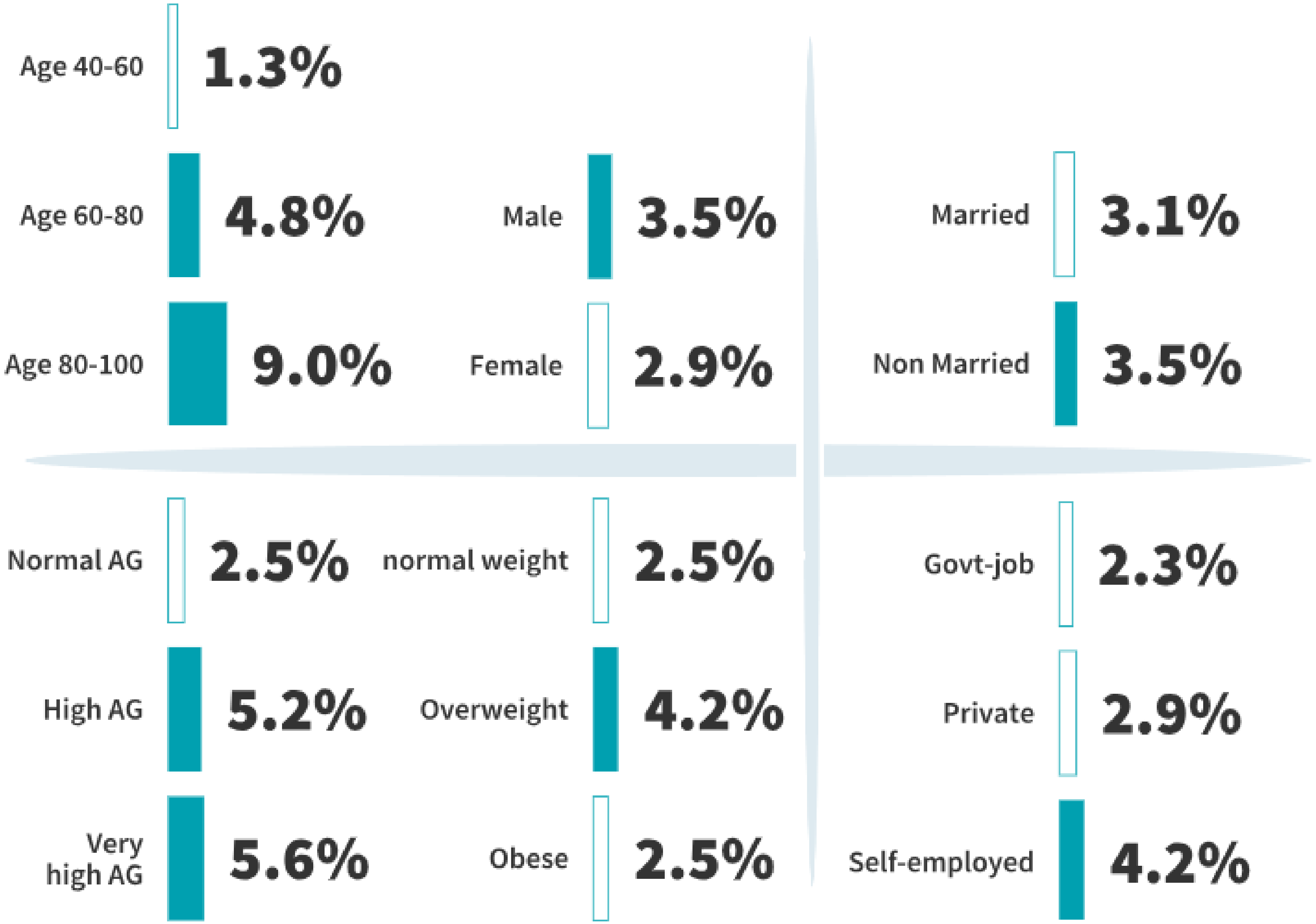
Male

Non Married

High Average Glucose
(>140 mg/l)

Overweight (but not obese)
30-50 BMI level

Self employed



Conducting the data preparation steps

1 Cleaning, binarizing, binning
Train & test split (70/30)

2 Standardization of the data, rebalancing of the data set (using Smote/Tomek)

3 Enrichment of the data through polynomial approach (121 attributes) then selection using Machine Learning (RFE) method, Random Forest, PCA

RESULT: Base Model Logistic regression baseline performance:
Stroke detection: 78% , False Alarm: 25%, Missed Stroke: 17.9%

Six models were compared to maximize Stroke Detection and minimize False alarm using Sklearn

1. **Logistic Regression (Baseline model)**
   "Probability of log-odd linear combination of the attributes"
   (+)Good for binary classification
   (-) Need clean data, no outliers

2. **Gaussian Naive Bayes**
   "Arg max of Naive Bayes probability model"
   (+) Simple. Works well on classification problems
   (-) Feature should not correlate with each other

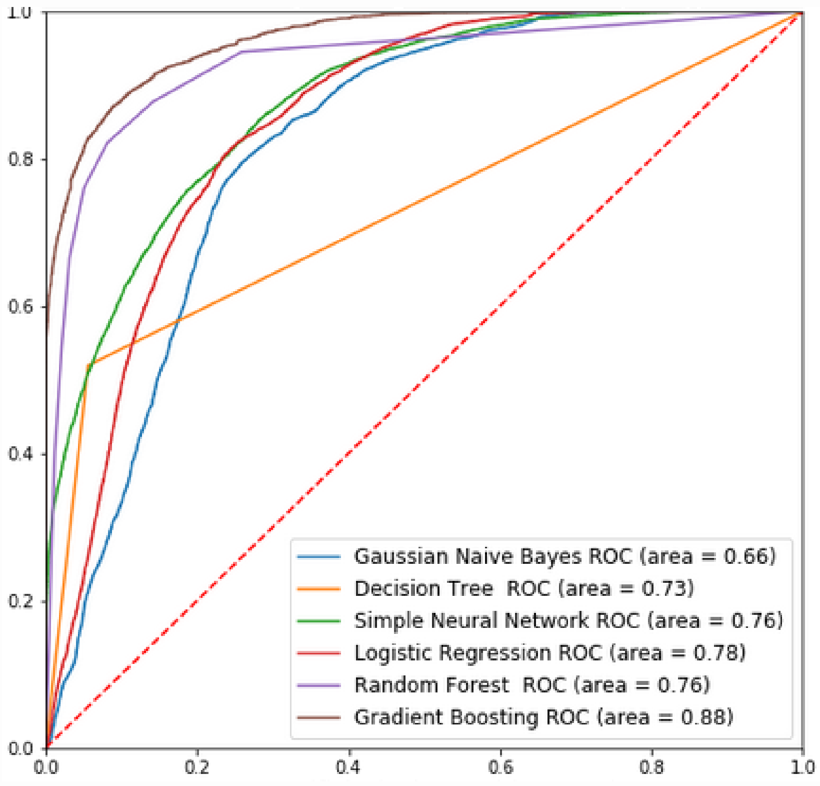
3. **Decision Tree**
   "Trees of decisions"
   (+) No data preparation, easy to understand
   (-) Works less well with binarized data

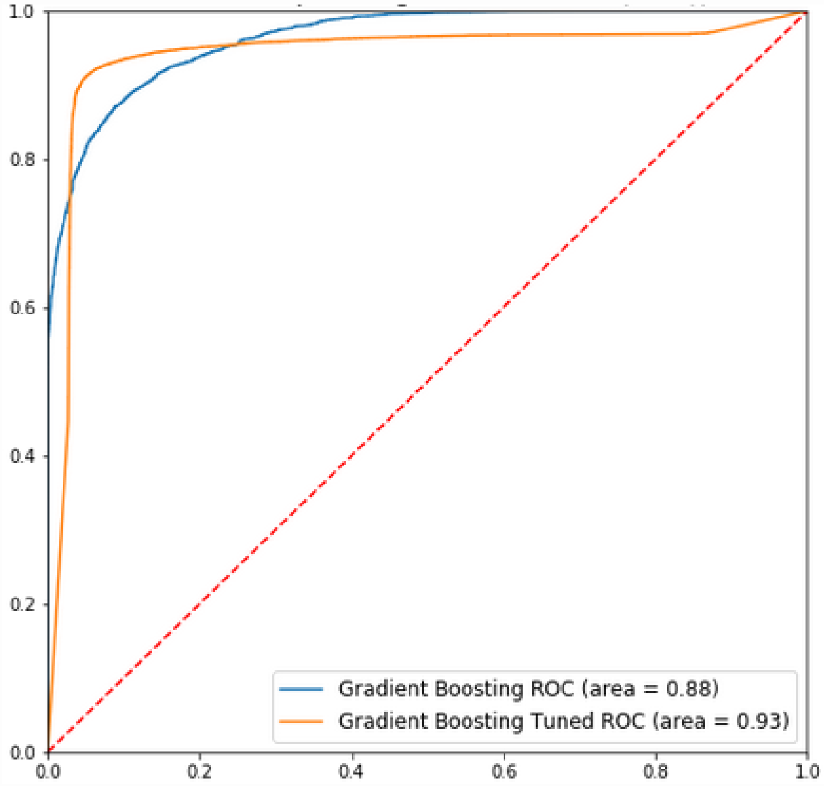
4. **Random Forest**
   "Forest of Decision trees"
   (+) Less susceptible to overfitting

5. **Gradient Boosting**
   "Random Forest + Use the gradient to correct the biggest mistakes and iterate"
   (+) Very Powerful
   (-) Must have clean data

6. **Simple Neural Network**
   " Mathematical models defining multiple transformation function"
   (+) No need to create features
   (-) Requires lots of data to avoid overfitting, black box

The Best algorithm (Detection/False Alarm) is...



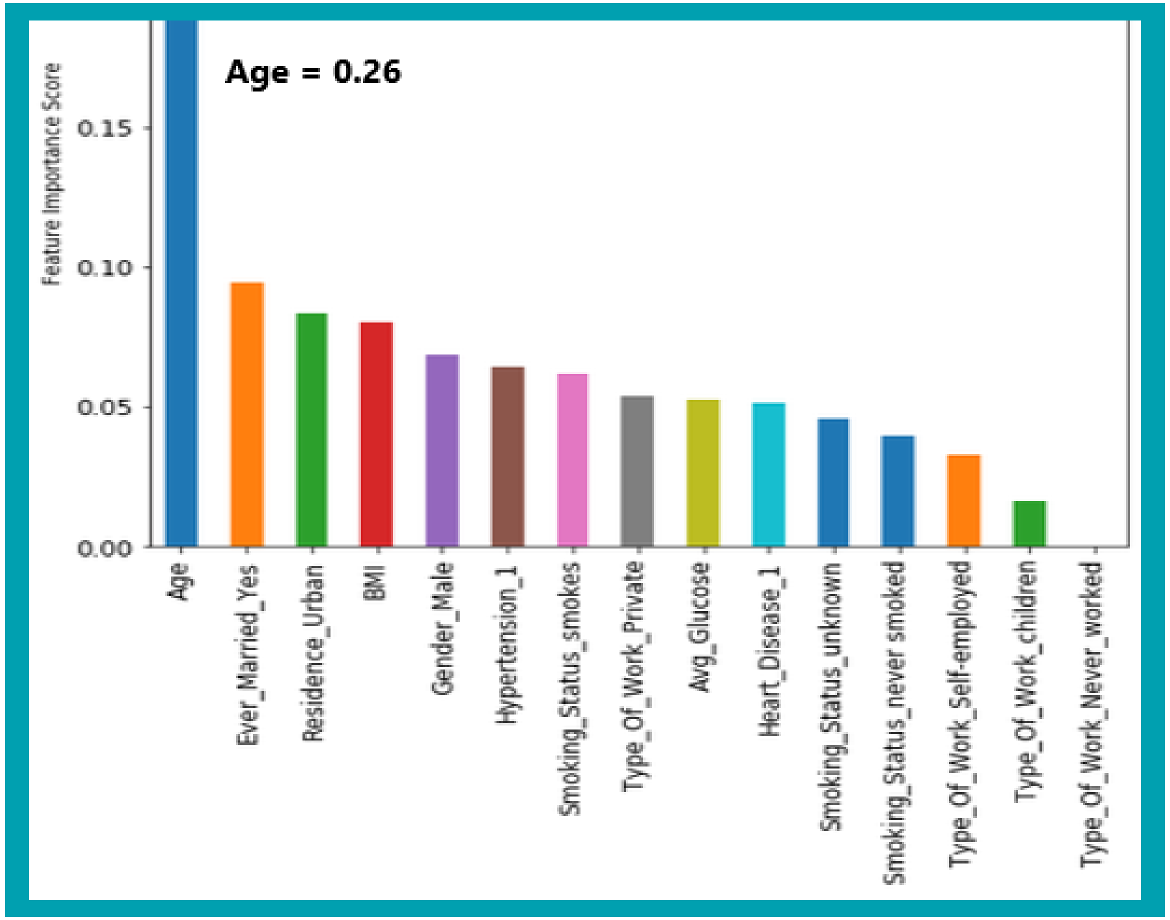


Gradient Boosting: 0.88 score + 13%

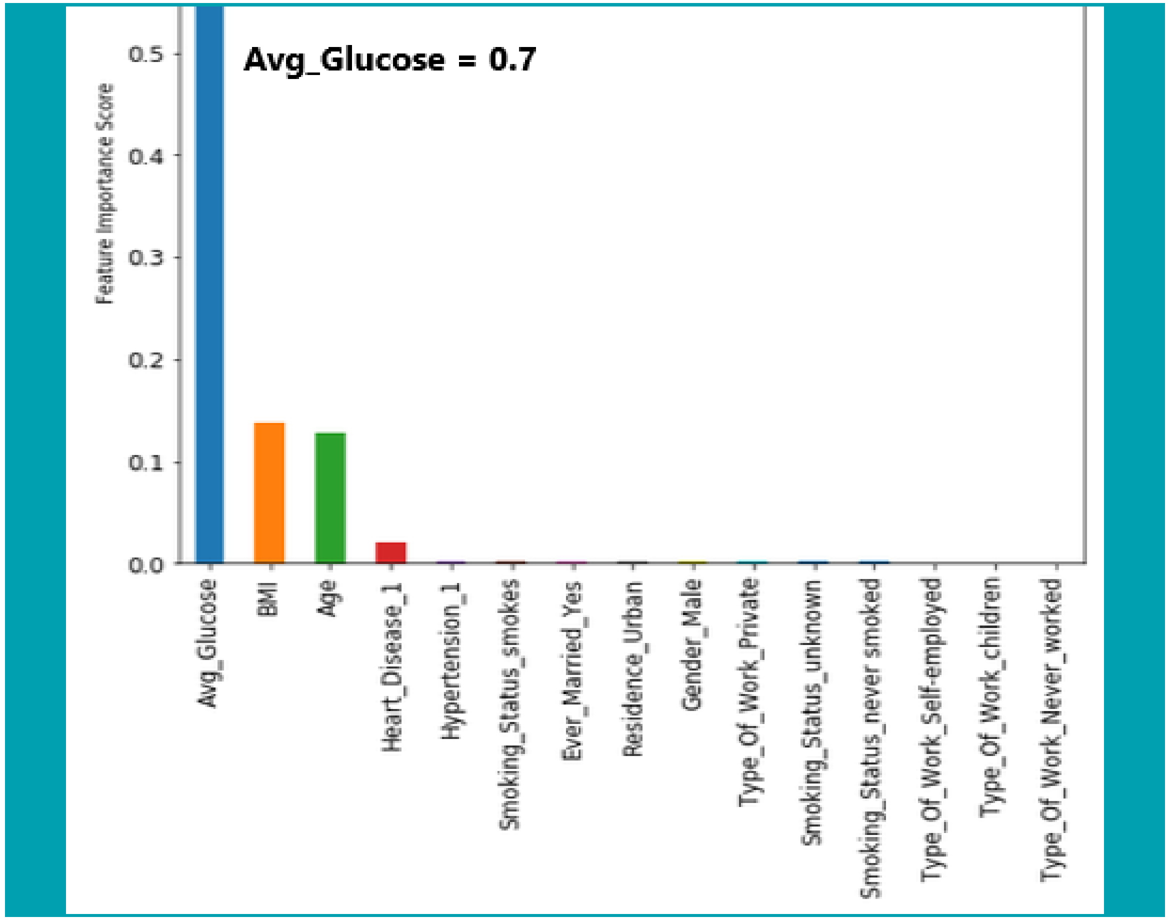
Gradient Boosting Tuned*: 0.93 score +19%

*learning_rate=1.0,n_estimators=600, min_samples_split=0.04, max_features=12, max_depth=8

Most important features...



Gradient Boosting: 0.88 score



Gradient Boosting Tuned*: 0.93 score

Next Steps

1

Quick Win: 2.5 days

Fine tune the threshold to minimize missed Strokes (increase detection and false alarm): 0.5 day

Re-run model with binned data to see if models improves: 1 day

Code documentation: 1 day

2

Data Correction: 4 days with VU hospital support

Correct data (ID, Smoking, BMI). Rerun predictions with Gradient Boosting method (H2o framework: mix data friendly algorithm) Could further improve performance by 1-2 basis point: 4 days

Option: Tune neural network to improve model. 1-2 basis point improvement: 2 days
Model explainability (using Lime, Shape) to help Doctors in their discussions with patients: 3 days

Outlook

1

Feasibility study: add more data in quantity: 5 days

Get at least 20%-30% more data in quantity to create a train/test/validation sample to increase the confidence with the model results

Atos could investigate getting more data in partnerships with other hospital in the Nethelands / in the EU and make recommendations.

2

Feasibility study: Enrich the Data: 5 days with VU hospital support

Predict the type of Strokes:

Ischemic, Transient Ischemic, Hemorrhagic

Capture lifestyle habits:

Eating, physical activity, drinking

Capture medical conditions:

Blood pressure, Cholesterol, Diabetes, Circulation problem, Arterial fibrillation

Outcome: recommendations.

Thank you!

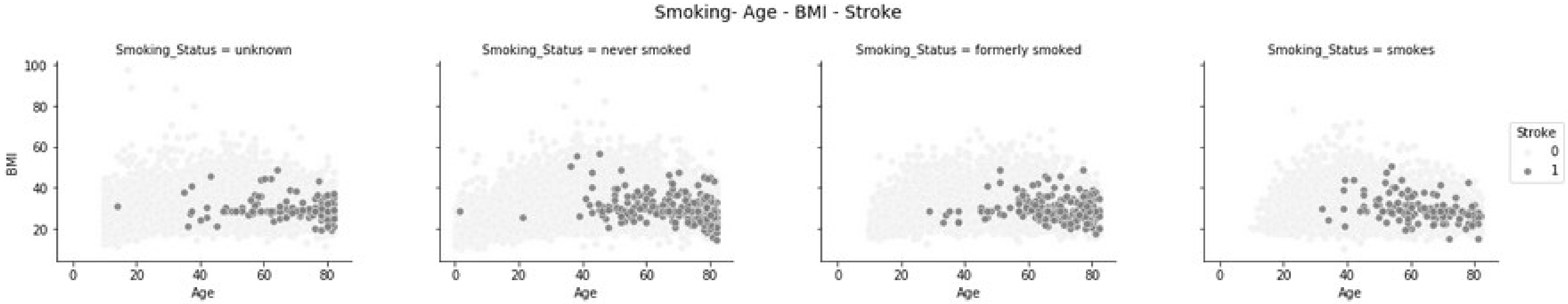
Appendix

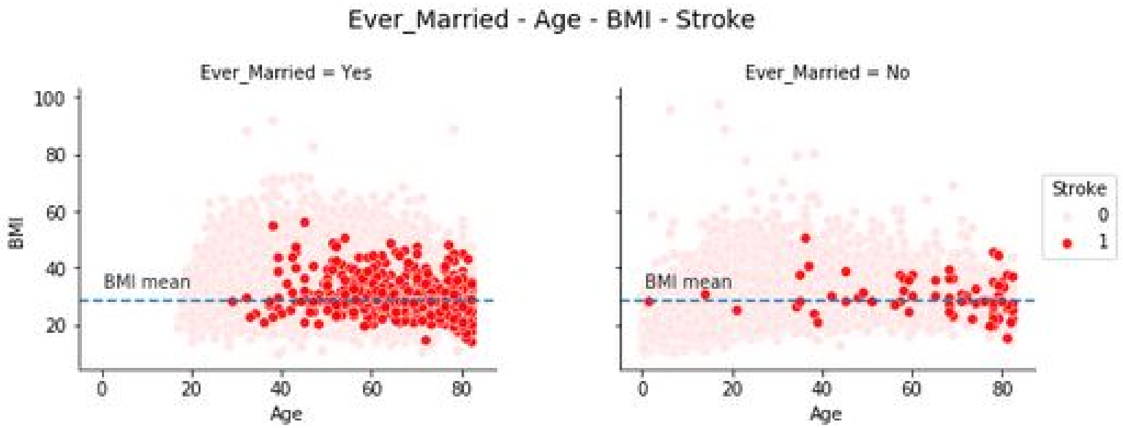
VU Medical Center Stroke Prediction Objectives

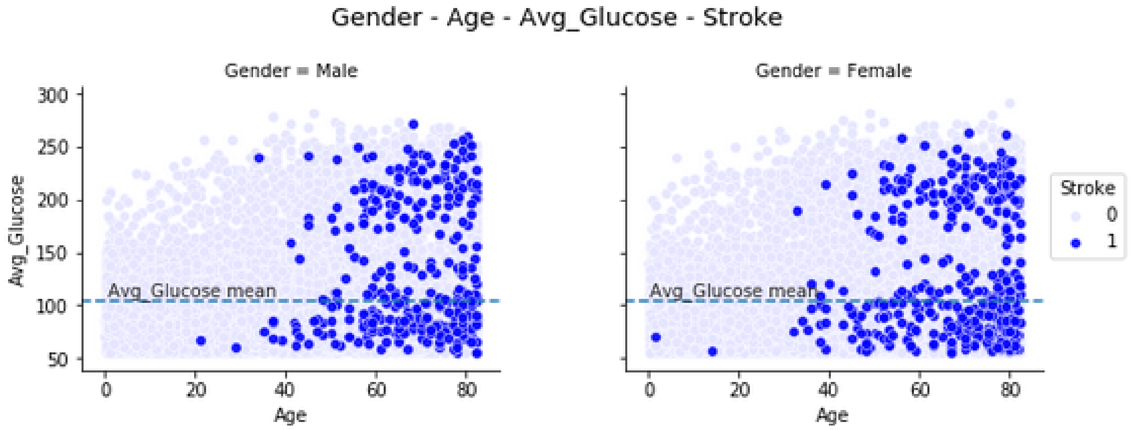
-Maximize detection of patients at risk of stroke with no MISS to SAVE LIVES
-Minimize FALSE ALARMS: visits of patients which are not needed to be efficient

PREDICTION OUTCOME	Predict: No Stroke	Predit: Stroke
Actual: No Stroke	SAVE TIME	FALSE ALARMS
Actual: Stroke	MISS	SAVE LIVES

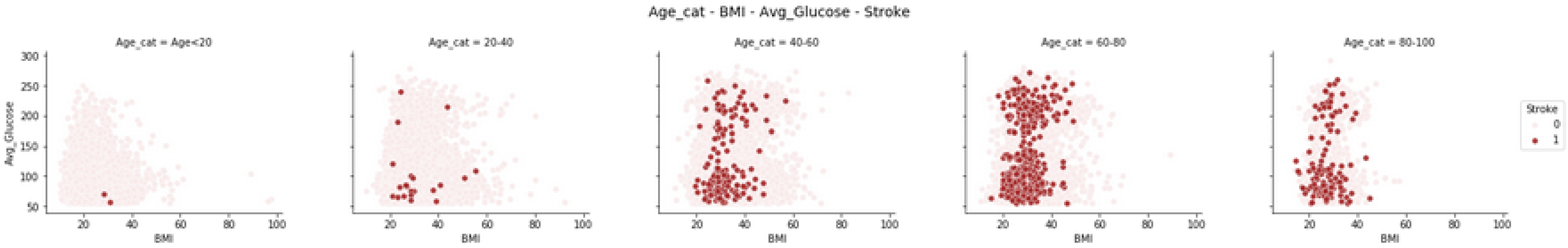
DETECTION RATE = SAVE LIVES / (SAVE LIVES+MISS)
FALSE ALARM RATE = FALSE ALARM/(SAVE LIVES+FALSE ALARM)

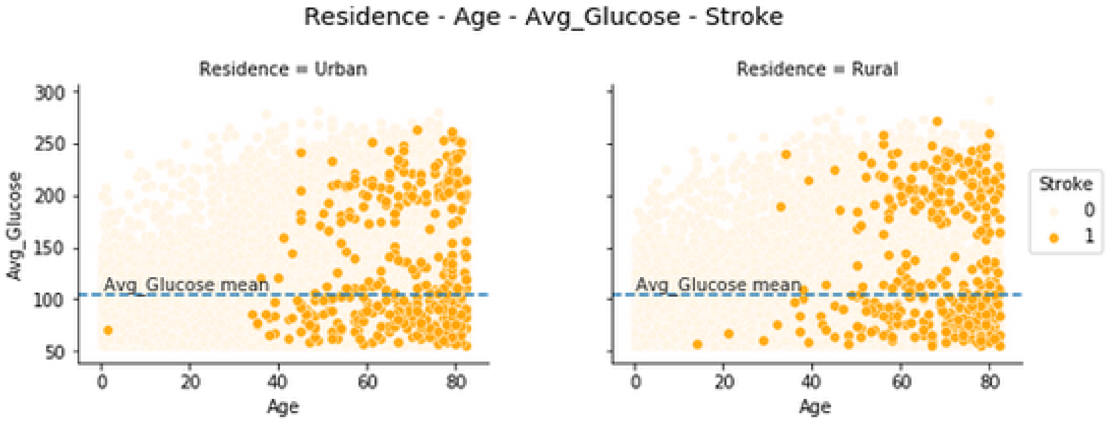


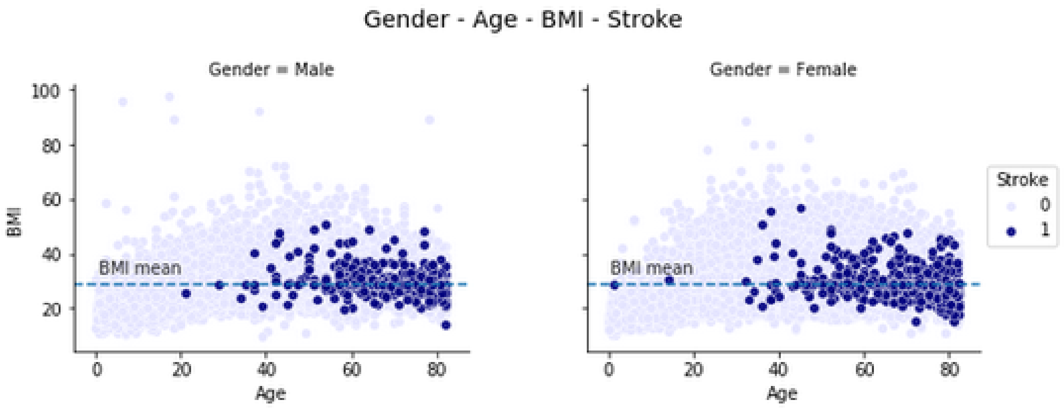




Exploration of the Data







Exploration of the data

```
Model Performance metrics:
------------------------------
Accuracy: 0.7843
Precision: 0.7858
Recall: 0.7843
F1 Score: 0.784

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.81      0.75      0.78     12755
          1       0.76      0.82      0.79     12755

avg / total       0.79      0.78      0.78     25510


Prediction Confusion Matrix:
------------------------------
          Predicted:
                0      1
Actual: 0    9539   3216
        1    2287  10468
```

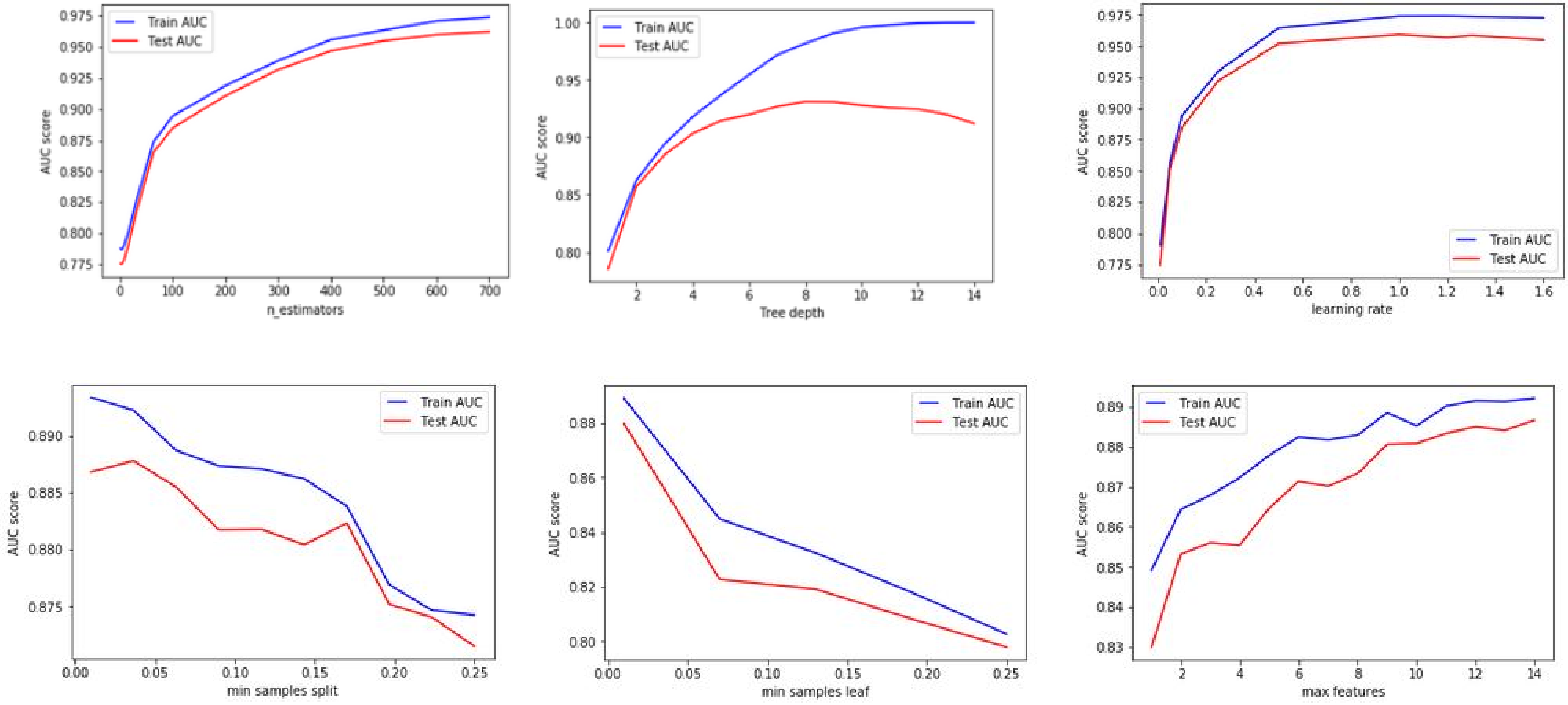
```
Model Performance metrics:
------------------------------
Accuracy: 0.7752
Precision: 0.7757
Recall: 0.7752
F1 Score: 0.7751

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.79      0.75      0.77     12755
          1       0.76      0.80      0.78     12755

avg / total       0.78      0.78      0.78     25510


Prediction Confusion Matrix:
------------------------------
          Predicted:
                0      1
Actual: 0    9629   3126
        1    2608  10147
```

Logistic Regression Algorithm: no improvements in performance with polynomial features (right)

```
Model Performance metrics:
------------------------------
Accuracy: 0.8519
Precision: 0.8536
Recall: 0.8519
F1 Score: 0.8517

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.88      0.82      0.85     12755
          1       0.83      0.89      0.86     12755

avg / total       0.85      0.85      0.85     25510


Prediction Confusion Matrix:
------------------------------
          Predicted:
                0      1
Actual: 0   10426   2329
        1    1449  11306
```

```
Model Performance metrics:
------------------------------
Accuracy: 0.9302
Precision: 0.9307
Recall: 0.9302
F1 Score: 0.9302

Model Classification report:
------------------------------
             precision    recall  f1-score   support

          0       0.92      0.95      0.93     12755
          1       0.95      0.91      0.93     12755

avg / total       0.93      0.93      0.93     25510


Prediction Confusion Matrix:
------------------------------
          Predicted:
                0      1
Actual: 0   12087    668
        1    1112  11643
```

Gradient Boosting Algorithm: clear performance improvement tuned (right) versus not tuned (left)



Optimization of Gradient Boosting Parameters