INTRODUCTION TO FOCUS AREA

# Project 3

Minie Jung, Kunaphas Kongkitimanon, Wan-Ju Chang

**Abstract**

**The goal of the project:** Getting the statistic information and developing models by using three different machine learning (clustering, classifier, and regression) to predict the accuracy and probability of stroke.

**The main result of the project:** We got the accuracy 76.83%, 83.98%, and 81.56% in Clustering, Predictor, and Classifier. Average probability scores we got are similar in Predictor(85.73%) and Classifier(85.73%).

**Keywords:** Predict the probability through machine learning; How to get the statistic information through SQL; Clustering; Predictor; Classifier

**An estimation of the time:** 16h

**Project evaluation:** 5

**The number of words:**
1325 (errors:1) words

## Scientific background and goal of the project

### Scientific background

A stroke is a rapid brain dysfunction caused by an imbalance in the blood supply to the brain. It happens when a blood vessel that carries oxygen and nutrients to the brain is blocked or interrupted. When that occurs, part of the brain cannot get the blood and oxygen; therefore, it dies. The significant factors of stroke include old age, high blood pressure, and smoking.

### Goal of the project

In task 1, we have to get statistics information by using the functions of spark SQL and process missing data. Through the regression model, we have to compare whether the effect of imputation processing influences the whole data analysis.

In task 2, we have to use three different machine learning (clustering, classifier, and regression) to apply in medical data. 1)clustering method: analyze the underlying structure of the data. 2)classifier: classify "no danger of stroke" vs. "danger of stroke." 3)regression analysis: predict the probability of a stroke happening.

## Description of the data

The dataset contains 62,001 patients and is divided into train data and test data. In train data, there is stroke information but not in test data. Also, there are missing values in both datasets.
Features :

1 id: Patient ID
2 gender: Gender of Patient
3 age: Age of Patient
4 hypertension : 0 - no hypertension, 1 - suffering from hypertension
5 heart_isease : 0 - no heart disease, 1 - suffering from heart disease
6 ever_rried: Yes/No
7 work_pe: Type of occupation
8 Residence_ype : Area type of residence (Urban/ Rural)
9 avg_lucose_evel: Average Glucose level (measured after meal)
10 bmi: Body mass index
11 smoking_tatus: patient's smoking status
12 stroke : 0 - no stroke, 1 - suffered stroke

### Summary of the data statistics

*Distribution*

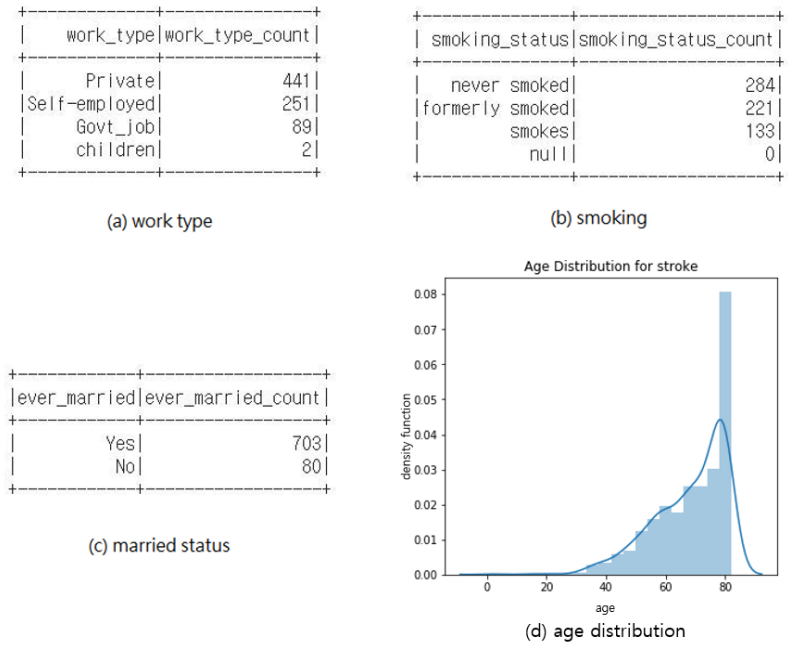


**Figure 1 Distribution analysis** (a)Stroke is mostly happening to a private and self-employed person. (b)Stroke is mostly happening to never smoked and formerly smoked person. (c)Stroke is mostly happening to a person who has ever been married. (d)Stroke is mostly happening to a person who is over 50 years old.

*Boxplot*

We made a boxplot(Figure:2) to observe outliers of features that have continuous values. In BMI and avg_glucous_level, we can detect many outliers.

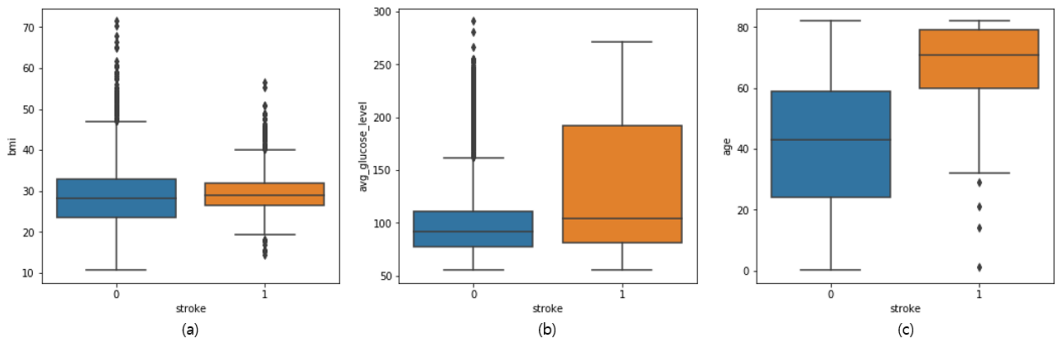


**Figure 2 Outliers analysis** (a)BMI and stroke (b)avg_glucose_level and stroke (c)age and stroke

*Correlation*

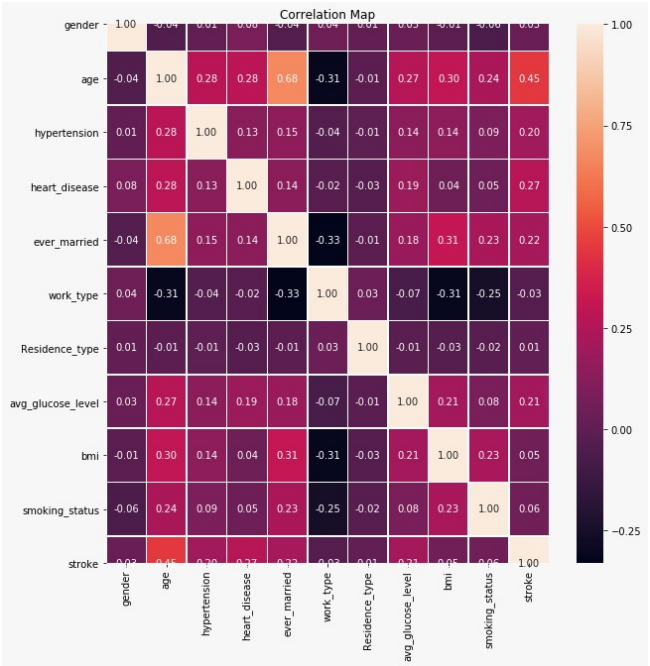


**Figure 3 Correlation analysis** Correlation values between features

The correlation values(Figure:3) between features and stroke are very low as we can see in this picture.

## Preprocessing

### Make dataset balanced

In the training dataset, 783 patients suffered a stroke and 42617 patients who are not suffered a stroke. The model could predict inaccurately with an imbalanced dataset, so we reduced non-stroke suffers to 3500 to make the dataset balanced.

### Goal of the project

We detected missing values in bmi and smoking_status and handled them with imputation and deletion; then, we can compare the result from 2 methods.

1 Imputation
Feature smoking_status has string values, so we imputed 'No info' to missing values of smoking_status. In feature bmi, we imputed the mean value of bmi column.

2 Deletion
In deletion, we deleted all rows which have missing values.

## Result

### Description of approaches

*Classifier*

We are using the Decision tree classifier and plug the algorithm into the pipeline. After the decision tree was built, we partition the training data set to train and validation to train the model and make the classification.

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
# Select (prediction, true label) and compute test error
acc_evaluator = MulticlassClassificationEvaluator(labelCol="stroke", predictionCol="prediction", metricName="accuracy")
dtc_acc = acc_evaluator.evaluate(dtc_predictions)
print('A Decision Tree algorithm had an accuracy of: {0:2.2f}%'.format(dtc_acc*100))
```

```
A Decision Tree algorithm had an accuracy of: 81.56%
```

**Figure 4 Accuracy of decision tree**

| | id | features | prediction |
|---|---|---|---|
| 0 | 36306 | (0.0, 1.0, 80.0, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0,... | 0.0 |
| 1 | 61829 | (1.0, 0.0, 74.0, 0.0, 1.0, 1.0, 0.0, 1.0, 0.0,... | 0.0 |
| 2 | 14152 | (1.0, 0.0, 14.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,... | 0.0 |
| 3 | 12997 | (0.0, 1.0, 28.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0,... | 0.0 |
| 4 | 40801 | (1.0, 0.0, 63.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0,... | 0.0 |
| 5 | 9348 | (1.0, 0.0, 66.0, 1.0, 0.0, 1.0, 1.0, 0.0, 0.0,... | 0.0 |
| 6 | 51550 | (1.0, 0.0, 49.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0,... | 0.0 |
| 7 | 60512 | (0.0, 1.0, 46.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0,... | 0.0 |
| 8 | 31309 | (1.0, 0.0, 75.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0,... | 1.0 |
| 9 | 39199 | (0.0, 1.0, 75.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0,... | 1.0 |

**Figure 5 Prediction result of decision tree**

*Predictor*

Logistic regression is a statistical method for predicting binary classes, which means the target variable is binary in nature (e.g. stroke=1/ non-stroke=0). We use this algorithm which utilizing a Sigmoid function to predict the probability of occurrence of a stroke event. In Logistic regression, it adopts Maximum Likelihood Estimation(MLE) to determine the parameters that are most likely to produce the observed data. This set of parameters can be used for predicting the data needed in a normal distribution.

| | id | prediction | probability |
|---|---|---|---|
| 0 | 36306 | 0.0 | [0.5714285714285714, 0.42857142857142855] |
| 1 | 61829 | 0.0 | [0.5714285714285714, 0.42857142857142855] |
| 2 | 14152 | 0.0 | [0.981089258698941, 0.018910741301059002] |
| 3 | 12997 | 0.0 | [0.981089258698941, 0.018910741301059002] |
| 4 | 40801 | 0.0 | [0.8723404255319149, 0.1276595744680851] |
| 5 | 9348 | 0.0 | [0.6666666666666666, 0.3333333333333333] |
| 6 | 51550 | 0.0 | [0.8723404255319149, 0.1276595744680851] |
| 7 | 60512 | 0.0 | [0.8723404255319149, 0.1276595744680851] |
| 8 | 31309 | 1.0 | [0.432258064516129, 0.567741935483871] |
| 9 | 39199 | 1.0 | [0.432258064516129, 0.567741935483871] |

**Figure 6 Prediction and probability of logistic regression**

*Predictor*

We use K-mean clustering as a clustering algorithm. We plug the algorithm into the pipeline. Since we knew that there were only two classes, so we set the target number of K (centroid) equals to two. After the model was built, we make predictions and evaluate clustering by computing the Silhouette score. The SquaredEuclideanSilhouette method computes the Euclidean distance over all the data of the dataset, which is a measure of how appropriately the data have been clustered.

```
+----+--------------------+----------+------+
|  id|            features|prediction|stroke|
+----+--------------------+----------+------+
|  38|(16,[0,2,6,10,11,...|       0.0|     0|
|  61|(16,[1,2,3,6,11,1...|       0.0|     0|
| 312|(16,[0,2,5,6,10,1...|       0.0|     0|
| 573|(16,[1,2,4,7,11,1...|       1.0|     1|
| 712|(16,[0,2,3,4,6,11...|       0.0|     1|
| 714|(16,[0,2,5,6,11,1...|       0.0|     1|
| 756|(16,[0,2,6,10,11,...|       0.0|     0|
| 919|(16,[0,2,3,5,7,11...|       0.0|     1|
|1010|(16,[1,2,8,11,12,...|       0.0|     0|
|1081|(16,[0,2,3,5,6,10...|       1.0|     1|
+----+--------------------+----------+------+
only showing top 10 rows
```

**Figure 7 The results of model prediction**

```
# Evaluate clustering by computing Silhouette score
evaluator = ClusteringEvaluator()

silhouette = evaluator.evaluate(kmean_predictions)
print("Silhouette with squared euclidean distance = " + str(silhouette))
```

```
Silhouette with squared euclidean distance = 0.8464796298282147
```

**Figure 8 Silhouette score**

```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator
# Select (prediction, true label) and compute test error
acc_evaluator = MulticlassClassificationEvaluator(labelCol="stroke", predictionCol="prediction", metricName="accuracy")
kmean_acc = acc_evaluator.evaluate(kmena_output_df)
print('A K-mean clustering algorithm had an accuracy of: {0:2.2f}%'.format(kmean_acc*100))
```

```
A K-mean clustering algorithm had an accuracy of: 76.80%
```

**Figure 9 Result of k-mean clustering**

For silhouette score, we obtained a pretty high value which indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

Moreover, we got around 76% of the clustering accuracy.

### Comparison of approaches

The probability, F1, and accuracy score for evaluation on both training/testing set come from average score.

| Method | Accuracy (%) | F1 score | Probability using independent test set (%) |
|---|---|---|---|
| Clustering (K-mean) | 76.83 | 0.76 | - |
| Predictor (Logistic Regression) | 83.98 | 0.83 | 85.73 |
| Classifier (Decision Tree) | 81.56 | 0.81 | 85.73 |

**Figure 10 Comparison table among methods with accuracy, F1 score, and probability**

When there is a large number of features with less noise, logistic regressions may outperform than decision trees. In general cases, both algorithms will have similar average probability score.

Clustering is a kind of unsupervised learning. Compared with the same type of supervised learning, the effect of unsupervised learning is worse because the model has no answer at the time of learning, so the information of misclassification data in the boundary of the classification will not be excellent. In our results, we can observe a similar situation. We got a lower accuracy score in clustering methods compared with supervised methods (predictor  classifier).

| Method | Key differences |
|---|---|
| Clustering (K-mean) | • All data is unlabeled<br>• Find natural clusters, where you are not teaching but it learns a structure or pattern in a collection of uncategorized data.<br>• Learning method takes place in real time. |
| Predictor (Logistic regression) | • Estimates the probability of class membership by using a multilinear function of the features.<br>• All data is labeled |
| Classifier (Decision Tree) | • All data is labeled<br>• Selecting the best attribute to divide a set at each branch and generate multiple decision boundaries. |

**Figure 11 The differences of methods**

In summary, both classifier and predictor may refer to the classification; however, they differ in the way that they generate decision boundaries. Contrarily, the clustering model tries to discover a structure from the input data, and present the interesting structure in the data.

## Effect of imputation

Comparison of the predictor trained with and without imputed data

```
p_mean = a/(predict_test_p.count().probability)
print('A Logistic Regression algorithm with imputation had prediction probability of {0:2.4f}%'.format(p_mean*100
```

```
A Logistic Regression algorithm with imputation had prediction probability of 85.7350% using independent test set
```

```
print("Coefficients: ", model_lr.stages[-1].coefficients)
print("Intercepts:", str(model_lr.stages[-1].intercept))
```

```
Coefficients:  [-1.596925742757241,-1.3555589065577358,0.07465284067507821,0.3528583778924133,0.9066054178632676,
Intercepts: -2.226285833066799
```

**Figure 12 Result of With imputation**

```
del_p_mean = a/(predict_test_del_p.count().probability)
print('A Logistic Regression algorithm without imputation had prediction probability of {0:2.4f}%'.format(del_p_mean
```

```
A Logistic Regression algorithm without imputation had prediction probability of 84.0559% using independent test set
```

```
print("Coefficients: ", model_del.stages[-1].coefficients)
print("Intercepts:", str(model_del.stages[-1].intercept))
```

```
Coefficients:  [-1.5343536529373494,-1.3544346532429141,0.07556839889038569,0.4097004751142307,0.637012764764908,-0.
Intercepts: -2.213966973129203
```

**Figure 13 Result of Without imputation (delete row)**

There are no significant difference between imputation and deletion based on accuracy of predicting. It because the correlation value between “stroke” and “bmi”, “stroke” and “smoking_status” are low as we can see from the correlation heatmap. Even though the difference is very small, the trained model with imputation show higher accuracy. The correlation value between “stroke” and two features becomes low after deletion, it can be a reason of difference between accuracy.

## Discussion

Why is this a typical project for a data-scientist?

Through this project, we learn how to get statistics information by utilizing the functions of spark SQL and process missing data. Through the effect of imputation testing in the regression model, we compare the analysis with imputations and without imputations, further to figure out whether it influences the whole data analysis.

In task 2, we learn to approach the data through three different machine learning (clustering, classifier, and regression). Through clustering method, it aids us to understand the underlying structure of the data. The classifier we used to classify "no danger of stroke" vs. "danger of stroke." In regression analysis, we learn the function of it to predict the probability of a stroke happening.

In this project, we utilize several ways to grasp the meaningful information from the data set. The pursuit of us in this project total matches the purpose of data scientists. Therefore, we think it is a typical project for data science.