ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ
UNIVERSITY OF PATRAS

Εξόρυξη Δεδομένων και Αλγόριθμοι Μάθησης
Project Python 2021

Ζερβός Νικόλαος 1054361
Φιλιππίδης Πρόδρομος 1046160

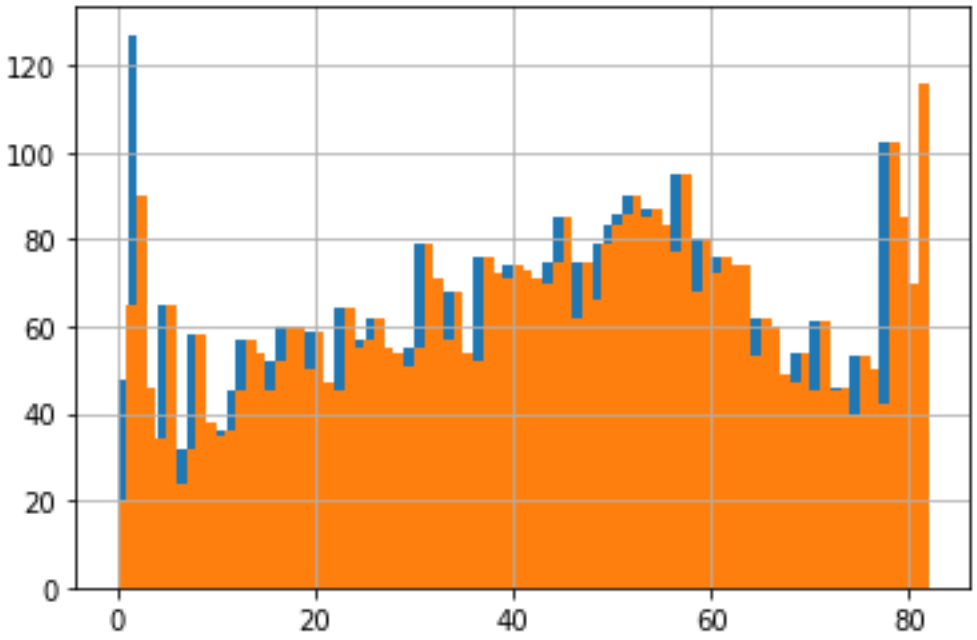
ΕΡΏΤΗΜΑ 1

Για το ερώτημα 1, λάβαμε ως αρχείο δεδομένων το *healthcare-dateset-stroke-data.csv*, το οποίο περιλαμβάνει πληροφορίες ασθενών, καθώς και το πόρισμα για το εάν έχουν υποστεί κάποιο εγκεφαλικό επεισόδιο ή όχι. Πιο συγκεκριμένα:

- Ηλικία

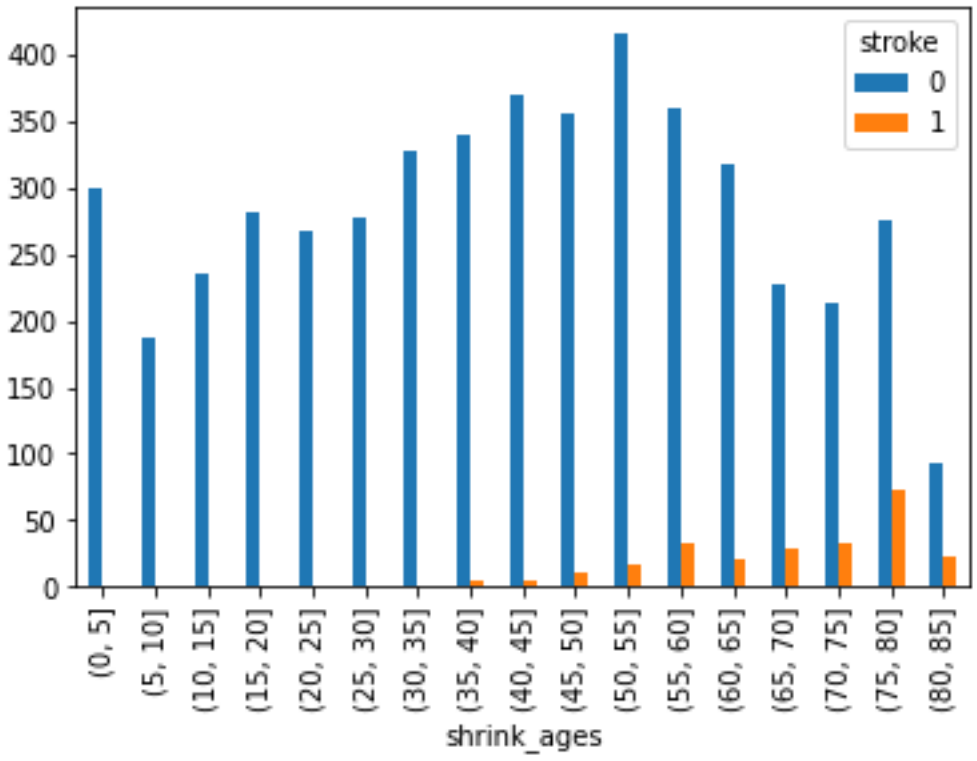
  Επεξεργαστήκαμε πρώτα το δεδομένο της ηλικίας. Αρχικά παρατηρήσαμε πως στις τιμές που λαμβάνουμε από την στήλη της ηλικίας, υπάρχουν δεκαδικές τιμές. Αποφασίσαμε να τις χειριστούμε ως ακέραιες τιμές, συνεπώς στρογγυλοποιήσαμε κάθε ηλικία που έφερε δεκαδικό ψηφίο.

```
Minimum age: 0.08
Maximum age: 82.0
```



Drawing 1: Μπλε: Πρίν την στρογγυλοποίηση. Πορτοκαλί: Μετά την στρογγυλοποίηση

Έπειτα, εκτυπώσαμε σε ένα γράφημα το πόσοι έχουν υποστεί εγκεφαλικό επεισόδιο ανάλογα της ηλικίας τους. Μετατρέψαμε τις ηλικίες σε bins των 5 στοιχείων, προκειμένου να είναι πιο εξωραϊσμένα τα δεδομένα μας.

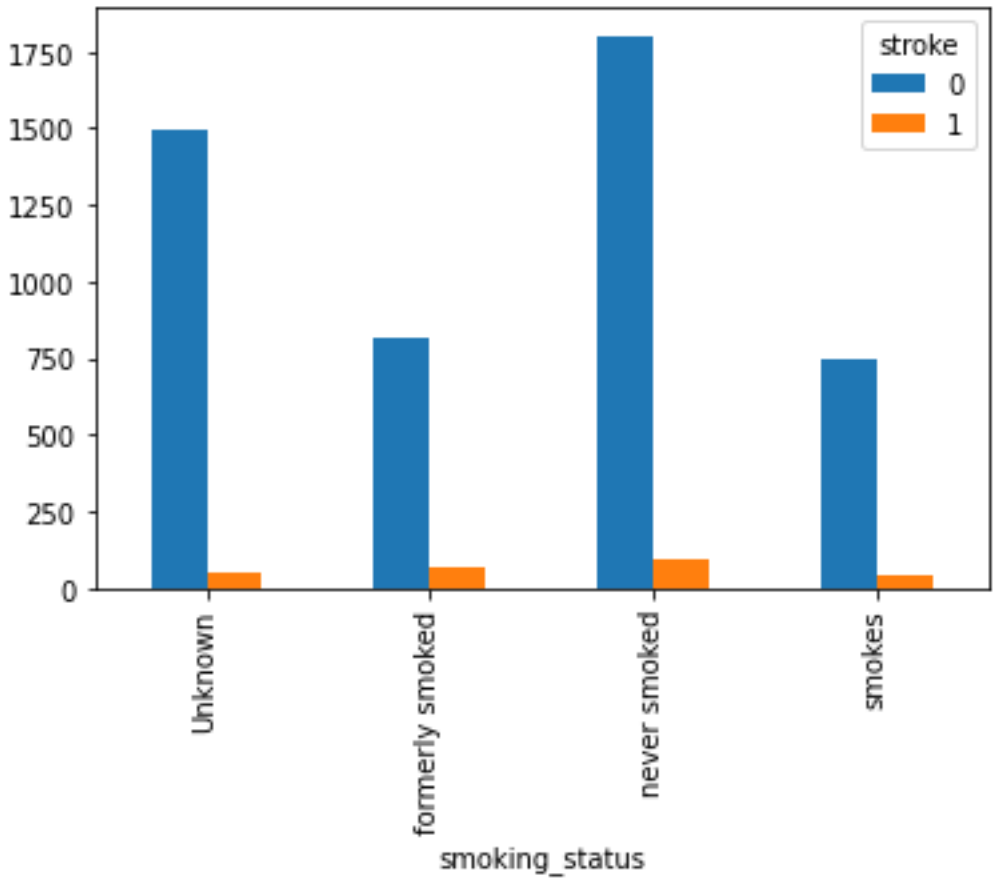


- Καπνιστική συνήθεια

  Στην συνέχεια αναλύσαμε την καπνιστική συνήθεια τον ασθενών στα δεδομένα μας και είδαμε τις εξής συμπεριφορές:

```
Smoking status
never smoked       1892
Unknown            1544
formerly smoked     885
smokes              789
Name: smoking_status, dtype: int64
```

Έπειτα, εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό επεισόδιο αναλόγως την καπνιστική τους συνήθεια.

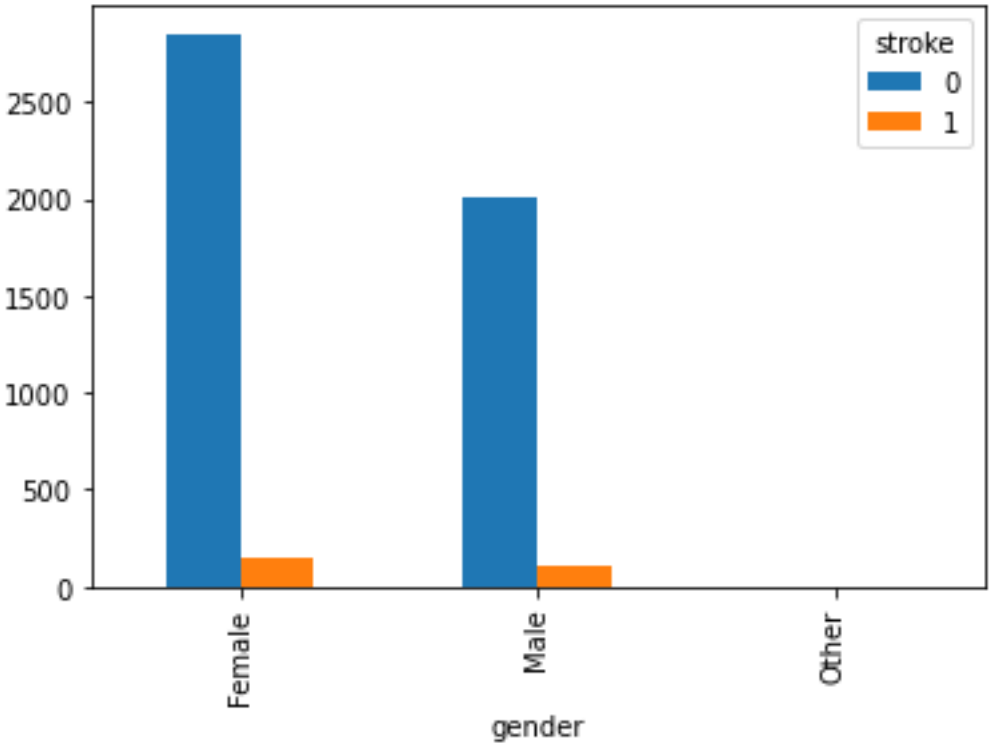


- Φύλο

  Ακολουθεί η ανάλυση του φύλου των ασθενών που εμπεριέχονται στα δεδομένα μας.

```
Genders
Female    2994
Male      2115
Other        1
Name: gender, dtype: int64
```

Έπειτα εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό επεισόδιο αναλόγως το φύλο τους.

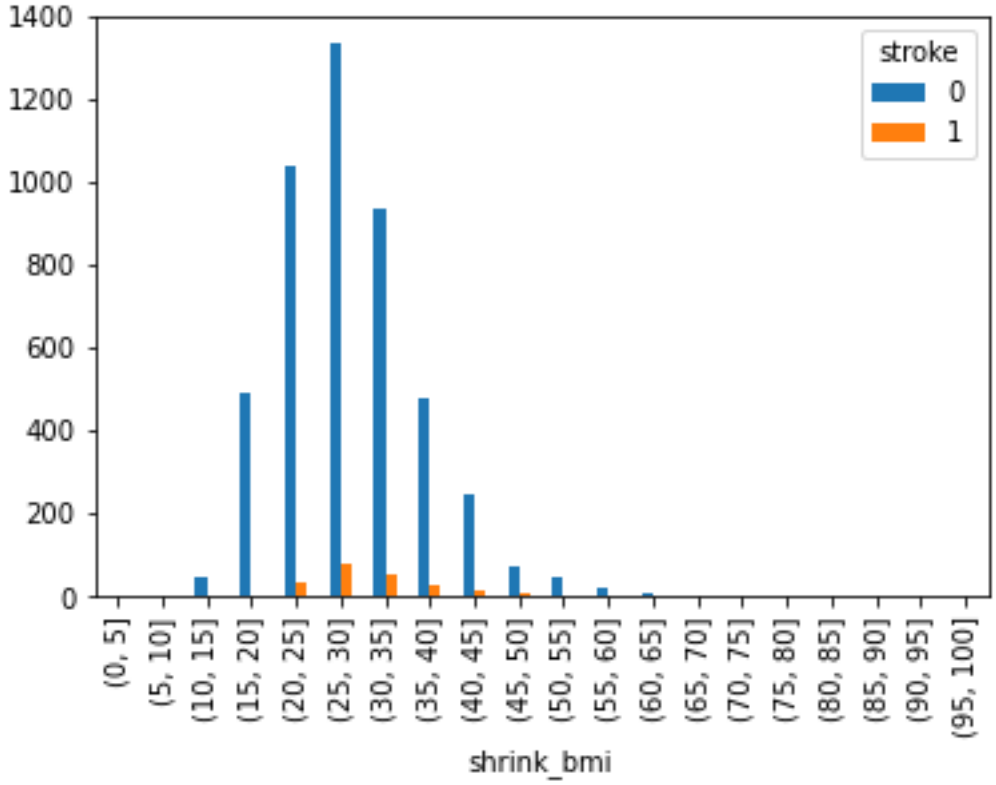


- BMI

  Σειρά έχει η ανάλυση του BMI (δείκτη μάζας) των ασθενών στα δεδομένα μας.

```
Minimum BMI: 10.3
Maximum BMI: 97.6
```

Έπειτα εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό ανάλογα του δείκτη μάζας τους. Για την πιο εξωραϊσμένη απεικόνιση του BMI, δημιουργήσαμε bins των 5 στοιχείων για να εμφανίσουμε τα δεδομένα μας.

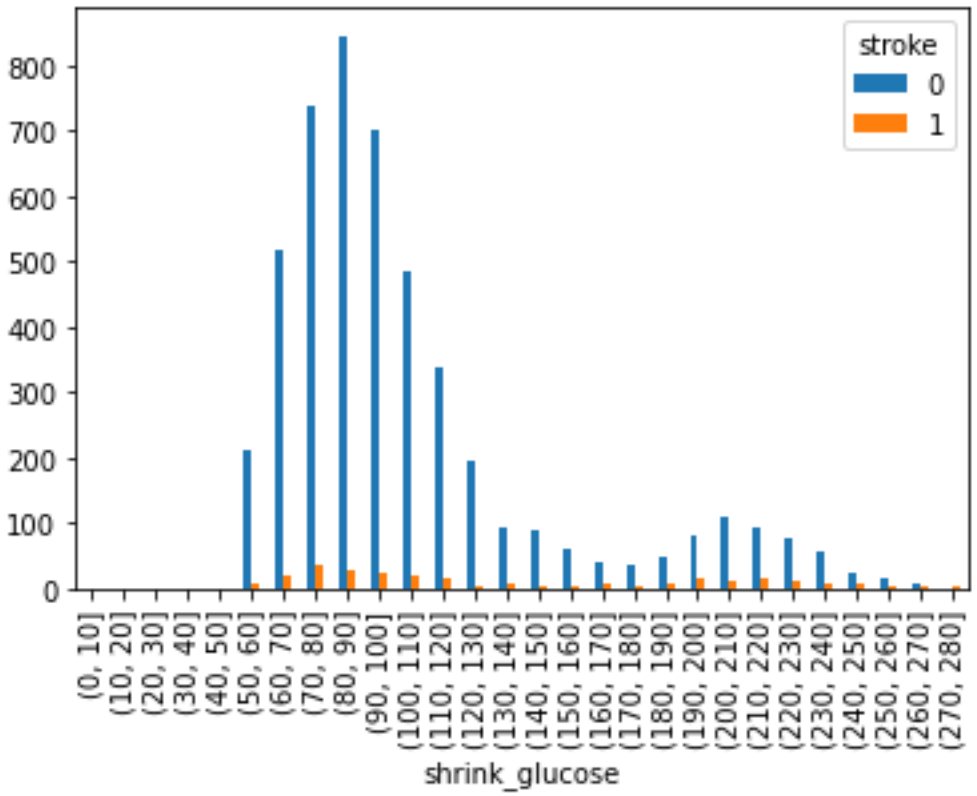


- <u>Επίπεδα γλυκόζης</u>

  Μετά ακολουθεί η ανάλυση των επιπέδων γλυκόζης των ασθενών στα δεδομένα μας.

```
Minimum average glucose level: 55.12
Maximum average glucose level: 271.74
```

Έπειτα εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό σε σχέση με τα επίπεδα γλυκόζης. Για την εξωραϊσμένη απεικόνιση των δεδομένων μας,

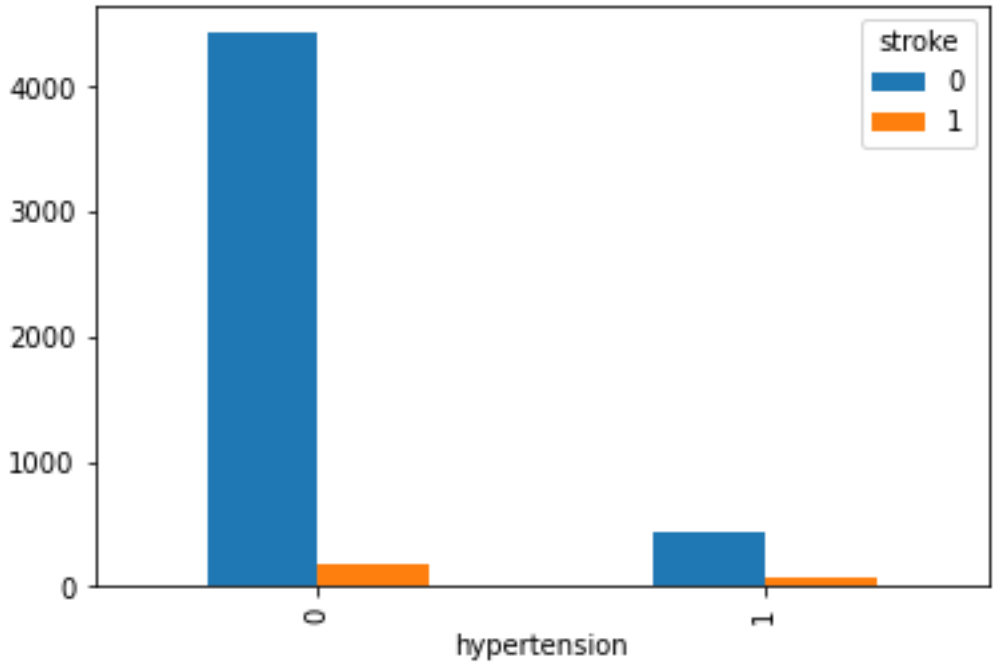

stroke
0
1
800
700
600
500
400
300
200
100
0
(0, 10]
(10, 20]
(20, 30]
(30, 40]
(40, 50]
(50, 60]
(60, 70]
(70, 80]
(80, 90]
(90, 100]
(100, 110]
(110, 120]
(120, 130]
(130, 140]
(140, 150]
(150, 160]
(160, 170]
(170, 180]
(180, 190]
(190, 200]
(200, 210]
(210, 220]
(220, 230]
(230, 240]
(240, 250]
(250, 260]
(260, 270]
(270, 280]
shrink_glucose

- Υπέρταση

  Εξετάσαμε στην συνέχεια το πόσοι ασθενείς πάσχουν από υπέρταση.

```
Hypertension
0    4612
1     498
Name: hypertension, dtype: int64
```

Έπειτα εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό σε σχέση με το εάν υποφέρουν από υπέρταση ή όχι.

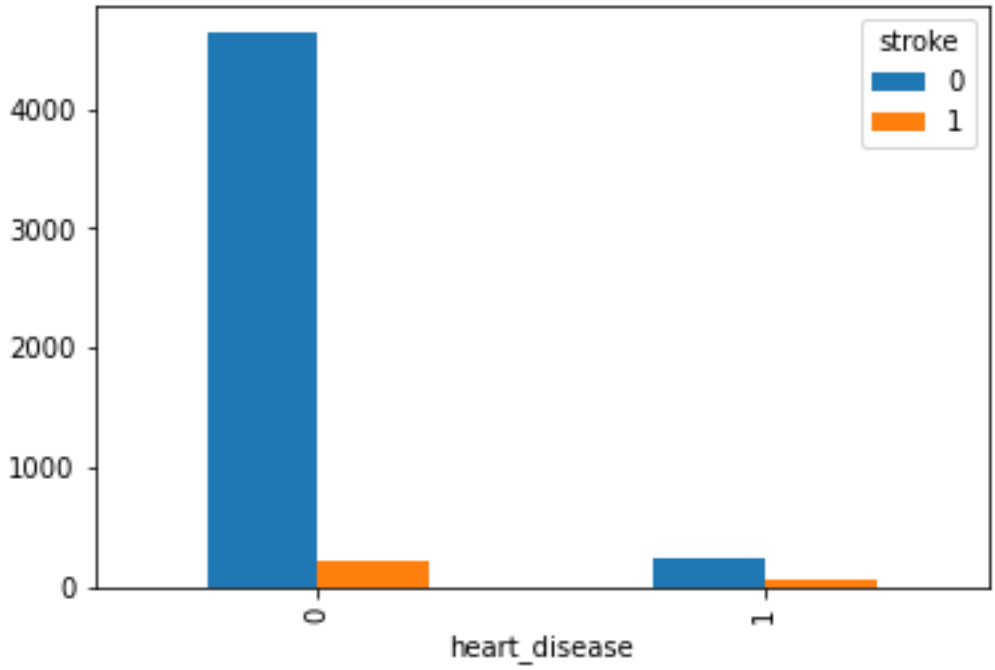


- Καρδιακή Πάθηση

  Επόμενο προς εξέταση δεδομένων είναι το εάν ο ασθενής πάσχει από κάποια καρδιακή πάθηση ή όχι.

```
Heart Disease
0    4834
1     276
Name: heart_disease, dtype: int64
```

Έπειτα εκτυπώσαμε σε γράφημα το πόσοι έχουν υποστεί εγκεφαλικό επεισόδιο ανάλογα εάν έχουν υποστεί κάποια καρδιακή πάθηση.

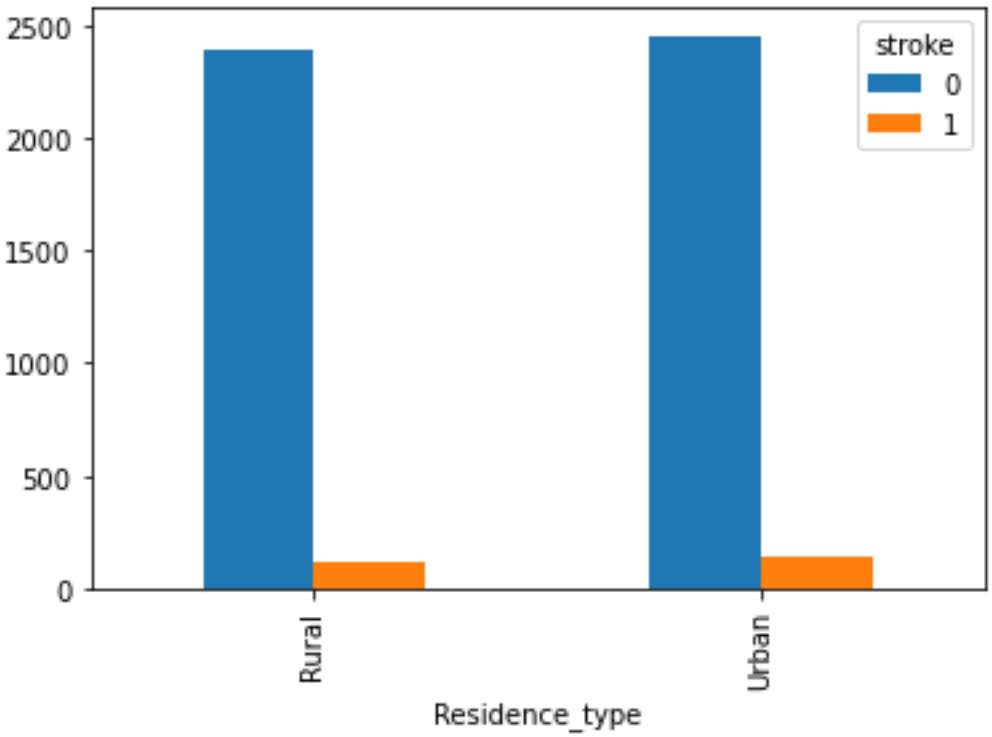


- <u>Περιβάλλον διαμονής</u>

  Επιπλέον προς ανάλυση δεδομένο είναι το περιβάλλον διαμονής των ασθενών.

```
Residence type
Urban    2596
Rural    2514
Name: Residence_type, dtype: int64
```

Έπειτα εκτυπώσαμε σε γράφημα το κατά πόσο επηρεάζει τον ασθενή το περιβάλλον διαμονής στο εάν έχει υποστεί εγκεφαλικό επεισόδιο ή όχι.

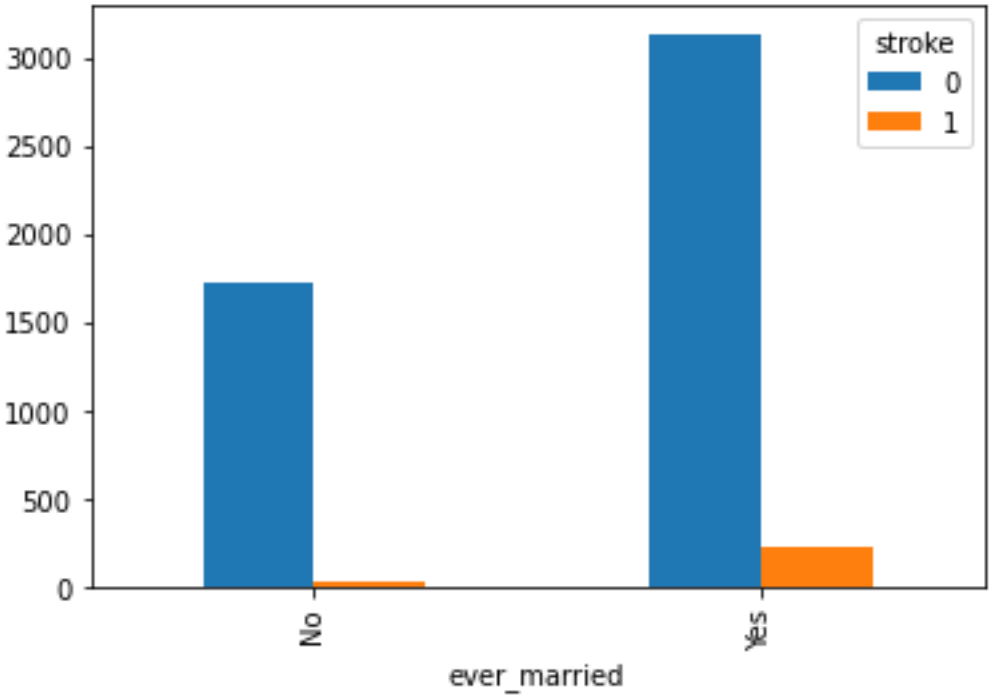


- <u>Οικογενειακή Κατάσταση</u>

  Προτελευταίο δεδομένο είναι η οικογενειακή κατάσταση των ασθενών που βρίσκονται στα δεδομένα μας.

```
Ever married
Yes    3353
No     1757
Name: ever_married, dtype: int64
```

Ακολουθεί το γράφημα που δείχνει την κατανομή των ασθενών σχετικά με το εάν έχουν υποστεί εγκεφαλικό επεισόδιο ή όχι ανάλογα με την οικογενειακή τους κατάσταση.

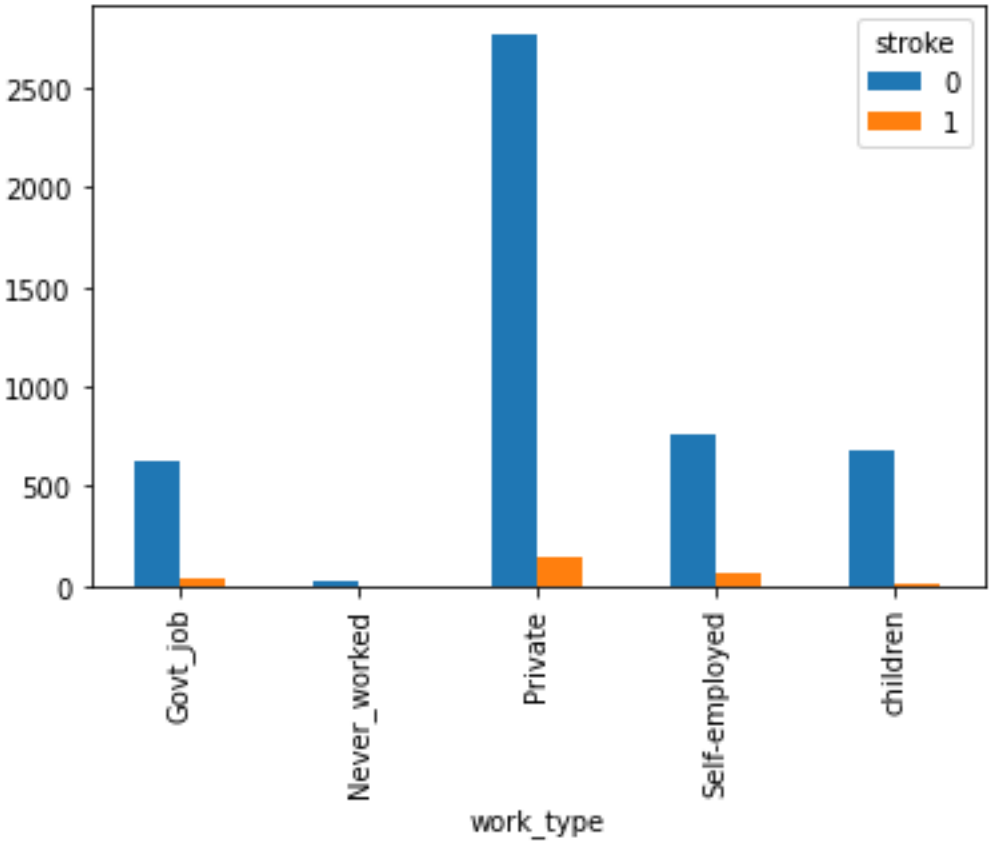


- <u>Φύση Εργασίας</u>

  Τέλος, αναλύουμε την φύση της εργασίας του κάθε ασθενή στα δεδομένα μας.

```
Work type
Private          2925
Self-employed     819
children          687
Govt_job          657
Never_worked       22
Name: work_type, dtype: int64
```

Τελευταίο έρχεται το γράφημα που δείχνει την κατανομή των ανθρώπων που έχουν υποστεί ή όχι εγκεφαλικό αναλόγως της φύσεως της εργασίας τους.



Μπορούμε πλέον, μετά την πλήρη απεικόνιση και μελέτη των δεδομένων μας, να προχωρήσουμε στην επεξεργασία τους.

Φτιάξαμε ένα αντίγραφο των δεδομένων μας, χωρίς τις περιττές για τους υπολογισμούς μας στήλες (όπως το id των ασθενών και τις συρρικνωμένες σε bins στήλες για την απεικόνιση των δεδομένων).

Χρησιμοποιήσαμε στην συνέχεια την κλάση LabelEncoder() για να μετατρέψουμε όλες τις τιμές των κατηγορικών σε δεδομένων σε αριθμητικές, όπως αυτές του φύλου, της οικογενειακής κατάστασης, της φύσεως της εργασίας και του περιβάλλοντος διαμονής, καθώς δεν θα μπορούσαμε να τις επεξεργαστούμε αλλιώς στην συνέχεια. Επιπλέον, κάναμε ειδική μεταχείριση στα δεδομένα της καπνιστικής συνήθειας, καθώς υπάρχουν τιμές ‘Unknown’, που τις θεωρήσαμε ως ελλειπή δεδομένα και τις μετατρέψαμε σε NaN. Έπειτα αναζητήσαμε όλες τις ελλιπείς τιμές στα δεδομένα μας και καταλήξαμε στις εξής:

```
Missing values from dataset
gender                 0
age                    0
hypertension           0
heart_disease          0
ever_married           0
work_type              0
Residence_type         0
avg_glucose_level      0
bmi                  201
smoking_status      1544
stroke                 0
dtype: int64
```

Για την διαχείριση των ελλιπών δεδομένων, χρησιμοποιήσαμε τις εξής τεχνικές:

1. Αφαίρεση στήλης (δημιουργία νέου συνόλου δεδομένων data_drop)

   Αφαιρέσαμε από όλο το σύνολο δεδομένων όποιο record είχε ελλιπή (NaN) τιμή. Συνολικά αφαιρέθηκαν 1684 records από τα δεδομένα μας.

2. Συμπλήρωση τιμών με τον μέσο όρο των στοιχείων της στήλης (δημιουργία νέου συνόλου δεδομένων data_replace_mean)

   Για τις στήλες καπνιστικής συνήθειας και δείκτη μάζας, εισάγαμε στην θέση των ελλιπών στοιχείων μας τον μέσο όρο της κάθε στήλης.
   Επιπλέον, για την καπνιστική συνήθεια στρογγυλοποιήσαμε τις τιμές που συμπληρώθηκαν με τον μέσο όρο, καθώς τα δεδομένα είναι ακέραιες τιμές από 0 έως 2, για να έχει νόημα η ετικετοποίηση που εφαρμόσαμε πιο πάνω.

3. Συμπλήρωση τιμών με χρήση Linear Regression (δημιουργία νέου συνόλου δεδομένων data_regression)

   Ξεκινάμε διαχωρίζοντας τις χρήσιμες για τον υπολογισμό του regression στήλες για το BMI, και στην συνέχεια κρατήσαμε εκείνα τα records όπου το BMI είναι NaN. Στην συνέχεια φτιάξαμε τα train και prediction σύνολα για να τα εισάγουμε στο Linear Regression Model (LinearRegression()). Έπειτα, οι τιμές που προβλέφθηκαν για τις NaN του BMI προστίθενται στο νέο μας σύνολο. Ακριβώς η ίδια μεθοδολογία ακολουθήθηκε και για το smoking status.

4. Συμπλήρωση τιμών με χρήση k-NN (δημιουργία νέου συνόλου δεδομένων data_knn)

   Χρησιμοποιήσαμε έναν imputer 5 γειτόνων που εφαρμόζει άμεσα τον αλγόριθμο k Nearest Neighbors. Έπειτα προχωρήσαμε σε στρογγυλοποίηση των τιμών του smoking status για έχει νόημα η ετικετοποίηση, όπως αναφέρθηκε και παραπάνω.

Για τα παραπάνω 4 νέα σύνολα δεδομένων που δημιουργήθηκαν, εφαρμόσαμε για το καθένα ξεχωριστά τον αλγόριθμο Random Forest, με αναλογία training – testing 75% - 25%. Επιπροσθέτως, στην στήλη stroke αυξήσαμε όλες τις τιμές κατά 1, καθώς πιθανή συγκέντρωση πολλών μηδενικών στα train-test sets δημιουργούσε πρόβλημα στον υπολογισμό των μετρικών. Έτσι για κάθε περίπτωση έχουμε για κάθε περίπτωση τα εξής αποτελέσματα:

- Για την αφαίρεση στήλης:

```
Accuracy:  94.632
F1 score:  0.97242206235012
Precision score:  0.9485380116959065
Recall score:  0.997539975399754
Confusion Matrix:
 [[811   2]
 [ 44   0]]
```

- Για την συμπλήρωση με τον μέσο όρο:

```
Accuracy:  95.540
F1 score:  0.9771726071285544
Precision score:  0.9553641346906813
Recall score:  1.0
Confusion Matrix:
 [[1220    0]
 [  57    1]]
```

- Για την συμπλήρωση με kNN:

```
Accuracy:  94.366
F1 score:  0.9710144927536233
Precision score:  0.945141065830721
Recall score:  0.9983443708609272
Confusion Matrix:
 [[1206    2]
 [  70    0]]
```

- Για την συμπλήρωση με Linear Regression:

```
Accuracy:  95.775
F1 score:  0.9784
Precision score:  0.9592156862745098
Recall score:  0.9983673469387755
Confusion Matrix:
 [[1223    2]
 [  52    1]]
```

ΕΡΩΤΗΜΑ 2

Αρχικά δημιουργούμε μια λίστα x , που κάθε στοιχείο της είναι ένα email από το dataset και ένα διάνυσμα y , που κάθε στοιχείο είναι η αντίστοιχη κατάσταση(spam = 1, no_spam = 0) για τα email της λίστας x. Στην συνέχεια χωρίζουμε σε training set(X_train, y_train) και test set(X_test, y_test) με αναλογία 75%-25% την λίστα x και το διάνυσμα y. Με την χρήση της κλάσης Tokenizer() καθώς και τις συναρτήσεις της fit_on_texts() και texts_to_sequence() κάθε λέξη σε ένα email αναπαριστάται από έναν πραγματικό αριθμό (η τιμή του πραγματικού αριθμού δείχνει την θέση που βρίσκεται η αντίστοιχη λέξη σε ένα λεξικό μεγέθους 19,542 λέξεων). Εφαρμόζοντάς το τώρα πάνω στα X_train, X_test δημιουργούμε μια λίστα που κάθε στοιχείο της είναι ένα διάνυσμα και με την χρήση της συνάρτησης pad_sequences(), βάζουμε μηδενικά (zero padding) στις αρχικές θέσεις κάθε διανύσματος έτσι ώστε το μήκος του κάθε διανύσματος της λίστας να είναι 200 και δημιουργούμε ένα μητρώο διαστάσεων length(X_train)x200 (το ίδιο κάνουμε και για το X_test).

Για την υλοποίηση του νευρωνικού δικτύου χρησιμοποιήσαμε ένα ακολουθιακό μοντέλο με 1^{ο} κρυφό επίπεδο ένα embedding layer με μήκος εισόδου 200 , μέγεθος λεξικού 19,542 και βάρη το μητρώο embedding_matrix που κάθε γραμμή του είναι το αντίστοιχο embedding vector για κάθε μία λέξη που υπάρχει στο αρχείο GloVe (glove.6B.200d). Για 2^{ο} κρυφό επίπεδο χρησιμοποιήσαμε ένα bidirectional LSTM με 75 νευρώνες για το Forward και το Backward Layer. Στο 3^{ο} κρυφό επίπεδο είχαμε 32 νευρώνες που κάθε νευρώνας είχε relu συνάρτηση ενεργοποίησης. Τέλος στο επίπεδο εξόδου είχαμε έναν νευρώνα με συνάρτηση ενεργοποίησης sigmoid καθώς μας έδωσε πολύ καλύτερα αποτελέσματα σε σχέση με την softmax στο binary classification πρόβλημά μας.

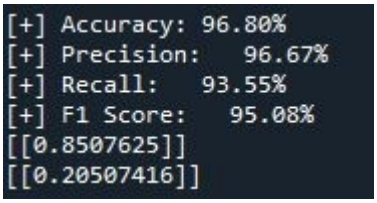
Στην συνέχεια εκπαιδεύουμε το δίκτυο για 10 εποχές με batch_size = 256 και validation_split = 0.25. Για το κομμάτι της αξιολόγησης περνάμε στο μοντέλο το X_test, y_test και παίρνουμε τις εξής μετρικές αξιολόγησης :

```
[+] Accuracy: 97.07%
[+] Precision:   95.93%
[+] Recall:   95.16%
[+] F1 Score:   95.55%
[[0.5444917]]
[[0.55409753]]
```

Οι τιμές που φαίνονται κάτω από τις μετρικές αξιολόγησης είναι τα αποτελέσματα που πήραμε χρησιμοποιώντας ένα μήνυμα που ήταν spam και ένα μήνυμα που δεν ήταν spam.

Ως spam ορίσαμε : *'Arabian prince won 5000 dollars, you are the only winner congrats claim now spam mail spam mail!'*
Ως not spam ορίσαμε : *'john remember to bring all the necessary files tomorrow at work please. It is very important and vital and crucial'*

Ακολούθως εκπαιδεύσαμε το δίκτυο για 20 εποχές και πήραμε τις παρακάτω μετρικές αξιολόγησης:



Παρόλο που στο κομμάτι των μετρικών δεν έχουμε ιδιαίτερες διαφορές, στο κομμάτι του prediction για το αν θα είναι spam ή no spam βλέπουμε ότι με αρκετά μεγάλη ακρίβεια έχει κάνει σωστή πρόβλεψη και ορθώς αποφασίζει ότι το πρώτο μήνυμα είναι spam ενώ το δεύτερο είναι no spam.

ΠΕΡΙΒΆΛΛΟΝ ΥΛΟΠΟΊΗΣΗΣ

Η ανάπτυξη του κώδικα έγινε σε γλώσσα Python 3.8. Χρησιμοποιήθηκε το IDE Spyder που παρέχεται από το Anaconda. Οι βιβλιοθήκες που χρησιμοποιήσαμε εγκαταστάθηκαν όλες από το Anaconda Navigator. Οι βιβλιοθήκες είναι οι εξής:

ΕΡΩΤΗΜΑ 1:

- *pandas*
- *numpy*
- *matplotlib*
- *sklearn*
  - *preproccessing*
    - *LabelEncoder*
  - *impute*
    - *KNNImputer*
  - *linear_model*
    - *LinearRegression*
  - *ensemble*
    - *RandomForestClassifier*
  - *model_selection*
    - *train_test_split*
  - *metrics*
    - *precision_score*
    - *accuracy_score*
    - *f1_score*
    - *recall_score*
    - *confusion_matrix*

ΕΡΩΤΗΜΑ 2:

- *tqdm*
- *numpy*
- *pandas*
- *keras*
  - *metrics*
    - *Precision*
    - *Recall*
  - *preproccessing*
    - *sequences*
      - *pad_sequences*
    - *text*
      - *Tokenizer*
  - *layers*
    - *Embedding*
    - *LSTM*
    - *Dense*
    - *Bidirectional*
  - *models*
    - *Sequential*
- *sklearn*
  - *model_selection*
    - *train_test_split*