Project Big Data Analytics

ΠΑΝΑΓΙΩΤΑ ΠΡΕΖΑ
ΑΜ:1089261
ΠΕΖ 2020

PART A

Στο συγκεκριμένο part χρησιμοποιήθηκε το σύνολο δεδομένων (clinical_data.csv) που περιέχει πληροφορίες που συλλέχθηκαν κατά την κλινική αξιολόγηση των ηλικιωμένων από ειδικούς γιατρούς. Αυτές οι πληροφορίες αντιπροσωπεύουν την κλινική κατάσταση του ηλικιωμένου σε διαφορετικούς τομείς (σωματικός, ψυχολογικός, γνωστικός κλπ.).

1. Preprocessing of the clinical dataset

Η γλώσσα προγραμματισμού που χρησιμοποίησα για την πραγματοποίηση του part A είναι η python και συγκεκριμένα οι βιβλιοθήκες pandas, numpy, sklearn.

Για την μετατροπή των κατηγορικών τιμών των χαρακτηριστικών σε αριθμητικές τιμές εντόπισα τις κατηγορικές τιμές των χαρακτηριστικών του clinical_dataset και με την χρήση της μεθόδου replace του data frame της βιβλιοθήκης pandas τις αντικατέστησα με αριθμούς (π.χ. Yes/No → 1/0, Frail / Pre-frail / Non-frail → 2 / 1 / 0, Hears well / moderate / poorly → 2 / 1 / 0) όπως φαίνεται στο παρακάτω κομμάτι κώδικα:

```
# Preprocessing of the clinical dataset
# Convert nominal features to numerical
df['fried'].replace(['Non frail','Pre-frail','Frail'],[0,1,2],inplace=True)
df['gender'].replace(['F','M'],[0,1],inplace=True)
df['ortho_hypotension'].replace(['No','Yes'],[0,1],inplace=True)
df['vision'].replace(['Sees poorly','Sees moderately','Sees
well'],[0,1,2],inplace=True)
df['audition'].replace(['Hears poorly','Hears moderately','Hears
well'],[0,1,2],inplace=True)
df['weight_loss'].replace(['No','Yes'],[0,1],inplace=True)
df['balance_single'].replace(['<5 sec','>5 sec'],[0,1],inplace=True)
df['gait_optional_binary'] = df['gait_optional_binary'].astype(int)
df['gait_speed_slower'].replace(['No','Yes'],[0,1],inplace=True)
df['grip_strength_abnormal'].replace(['No','Yes'],[0,1],inplace=True)
df['low_physical_activity'].replace(['No','Yes'],[0,1],inplace=True)
df['memory_complain'].replace(['No','Yes'],[0,1],inplace=True)
df['sleep'].replace(['No sleep problem','Occasional sleep
problem','Permanent sleep problem'],[0,1,2],inplace=True)
df['living_alone'].replace(['No','Yes'],[0,1],inplace=True)
df['leisure_club'].replace(['No','Yes'],[0,1],inplace=True)
df['house_suitable_participant'].replace(['No','Yes'],[0,1],inplace=True)
df['house_suitable_professional'].replace(['No','Yes'],[0,1],inplace=True)
df['health_rate'].replace(['1 - Very bad','2 - Bad','3 - Medium','4 -
Good','5 - Excellent'],[0,1,2,3,4],inplace=True)
df['health_rate_comparison'].replace(['1 - A lot worse','2 - A little
worse','3 - About the same','4 - A little better','5 - A lot
better'],[0,1,2,3,4],inplace=True)
df['activity_regular'].replace(['No','< 2 h per week','> 2 h and < 5 h per
week','> 5 h per week'],[0,1,2,3],inplace=True)
df['smoking'].replace(['Never smoked','Past smoker (stopped at least 6
months)','Current smoker'],[0,1,2],inplace=True)
```

Για την αφαίρεση των λανθασμένων τιμών (999 και test non applicable/adequate, test non realizable) χρησιμοποίησα την μέθοδο replace του data frame της βιβλιοθήκης pandas,αφού πρώτα τις είχα εντοπίσει ,όπως φαίνεται στο παρακάτω κομμάτι κώδικα.

```
# Remove erroneous values
missing_values=[999,'test non realizable','Test not adequate']
df.replace(missing_values,'',inplace=True)
```

Για την αντιμετώπιση των missing values που δημιουργήθηκαν από τα προηγούμενα ερωτήματα ή προϋπήρχαν στο dataset χρησιμοποίησα την στρατηγική της αφαίρεσης εγγραφών με missing values σε μερικά χαρακτηριστικά, κρίνοντας μετά από δοκιμές και των άλλων στρατηγικών ότι αυτή ταιριάζει καλύτερα στο μοντέλο πρόβλεψης. Συγκεκριμένα χρησιμοποίησα την μέθοδο replace του data frame της βιβλιοθήκης pandas μετατρέποντας τα missing values σε NaN και έπειτα χρησιμοποίησα την συνάρτηση dropna ώστε να αφαιρεθούν οι missing values,όπως φαίνεται στο παρακάτω κομμάτι κώδικα.

2. Classification

Έχοντας κάνει preprocessing στο δοθέν dataset εφάρμοσα αλγόριθμους κατηγοριοποίησης για την πρόβλεψη της παραμέτρου “fried” χωρίς να περιλαμβάνουν 5 παραμέτρους γενίκευσης (weight_loss, exhaustion_score, gait_speed_slower, grip_strength_abnormal, low_physical_activity). Συγκεκριμένα όπως φαίνεται στο παρακάτω κομμάτι κώδικα χρησιμοποίησα την συνάρτηση drop του data frame της βιβλιοθήκης pandas ώστε να αφαιρέσω από το dataset που έχει προκύψει από την προ επεξεργασία την παράμετρο πρόβλεψης “fried” καθώς και τις 5 προαναφερόμενές παραμέτρους που δεν πρέπει να συμπεριλαμβάνονται στο dataset. Έπειτα ανάθεσα το data frame της παραμέτρου “fried” στην μεταβλητή y και χώρισα το dataset X_train ,X_test,y_train,y_test φορτώνοντας την βιβλιοθήκη sklearn.model_selection την συνάρτηση train_test_split.Στην συνέχεια για να χρησιμοποιήσω τον classifier KNN φόρτωσα από την βιβλιοθήκη sklearn την συνάρτηση KNeighborsClassifier και αρχικοποίησα τον κατηγοριοποιητή θέτοντας 5 γείτονες. Χρησιμοποίησα την συνάρτηση fit ώστε να εκπαιδεύσω το νέο μας μοντέλο, προσαρμόζοντας τα δεδομένα εκπαίδευσης στο μοντέλο. Για την πρόβλεψη του μοντέλου χρησιμοποίησα την συνάρτηση predict στο Xtest και τέλος για την αξιολόγηση του μοντέλου φόρτωσα από την βιβλιοθήκη sklearn την συνάρτηση metrics ώστε να υπολογιστεί το accuracy του μοντέλου.

Με την ίδια λογική χρησιμοποιώντας το ίδιο dataset που έχει προκύψει από την προ επεξεργασία και έχοντας ήδη χωρίσει το dataset σε training και test σετ εφάρμοσα τον κατηγοριοποιητή Random Forest,φορτώνοντας από την βιβλιοθήκη sklearn.ensemble την συνάρτηση Random Forest Classifier θέτοντας ως n_estimators=25, χρησιμοποίησα την συνάρτηση fit ώστε προσαρμόσω τα δεδομένα εκπαίδευσης στο μοντέλο. Για την πρόβλεψη του μοντέλου χρησιμοποίησα την συνάρτηση predict στο Xtest και τέλος για την αξιολόγηση του μοντέλου είχα ήδη φορτώσει από την βιβλιοθήκη sklearn την συνάρτηση metrics ώστε να υπολογιστεί το accuracy του μοντέλου.

```
# Classification with KNN
# Drop fried and 5 parameters used for generating the fried
categorization
X =
df.drop(columns=['fried','weight_loss','exhaustion_score','gait_speed
_slower','grip_strength_abnormal','low_physical_activity'])

# Assign fried column to y
y = df['fried'].values

# Split dataset to train and test sets with 20% ratio
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.20, random_state=1)
# Initialize KNN classifier with number of neighbors to 5
knn = KNeighborsClassifier(n_neighbors = 5)
# Fit the k-nearest neighbors classifier from the training dataset
knn.fit(X_train,y_train)
# Make the prediction
y_pred = knn.predict(X_test)
# Model Accuracy
accuracy = float("{:.2f}".format(metrics.accuracy_score(y_test,
y_pred) * 100))
print("KNN Accuracy:",accuracy, "%")

#classsification with random forest
clf=RandomForestClassifier(n_estimators=25,random_state=1)
#Fit the random forest classifier from the training dataset
clf.fit(X_train,y_train)
# Make the prediction
y_pred=clf.predict(X_test)
# Model Accuracy
accuracy = float("{:.2f}".format(metrics.accuracy_score(y_test,
y_pred) * 100))
print("RF Accuracy:",accuracy, "%")
```

ΣΥΚΡΙΣΗ ΤΩΝ ΔΥΟ CLASSIFIERS

Το accuracy του KNN είναι 65.38% και του Random forest είναι 73.08% επομένως συμπεραίνουμε ότι η ακρίβεια του μοντέλου είναι καλύτερη όταν χρησιμοποιούμε τον Random Forest Classifier.Ενώ η ταχύτητα του KNN είναι 0.015sec καλύτερη από την ταχύτητα του Random Forest που είναι 0.055 sec κάτι αναμενόμενο καθώς έχουμε θέσει n_estimators=25 επιτυγχάνοντας καλύτερη ακρίβεια, όπως φαίνεται παρακάτω:

```
KNN Accuracy: 65.38 %
0.015620946884155273 seconds
RF Accuracy: 73.08 %
0.05571246147155762 seconds
```

Επομένως συμπεραίνουμε ότι ο KNN classifier είναι καλύτερος από θέμα χρόνου ενώ ο Random Forest Classifier είναι καλύτερος από θέμα ακρίβειας.