Diabetic patients Dataset

Python for Data analysis

Imen OULED DLALA

- By: Arnaud DE BRITO
- Nicolas FERRARA
- Constance LE FOURN

Problematic:

How to predict the readmission of diabetic patients

Libraries used:

-Numpy
-Pandas
-Matplotlib
-Seaborn
-Scikit-learn

Data preprocessing:

Let’s have a look to our data:

	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	...	citoglipton	insulin	glyburide-metformin
0	2278392	8222157	Caucasian	Female	[0-10)	?	6	25	1	1	...	No	No	No
1	149190	55629189	Caucasian	Female	[10-20)	?	1	1	7	3	...	No	Up	No
2	64410	86047875	AfricanAmerican	Female	[20-30)	?	1	1	7	2	...	No	No	No

Data preprocessing:

Then we have cleaned up the dataset. It includes encoding, dropping NaN and duplicates values.

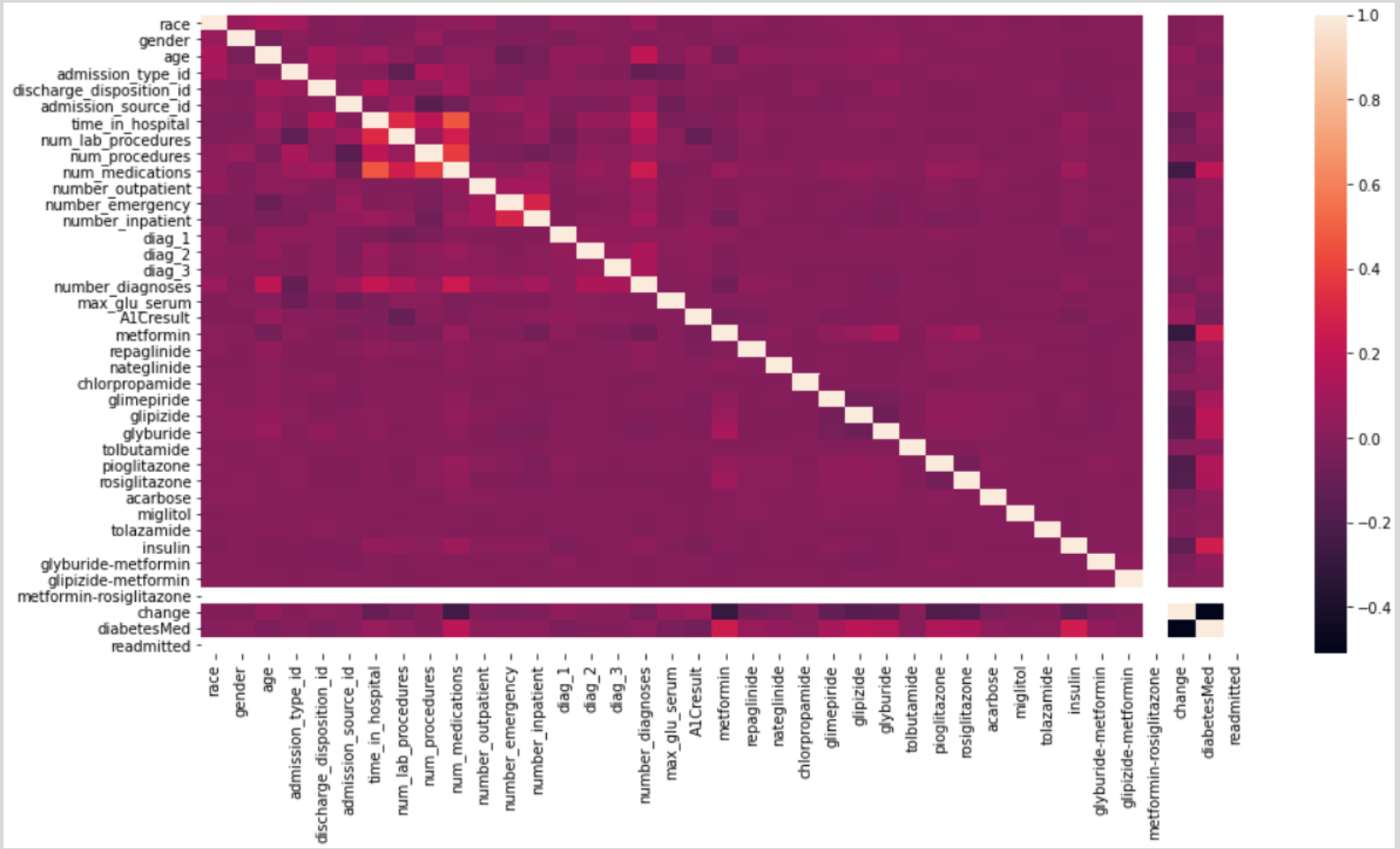
#	Column	Non-Null Count	Dtype
0	encounter_id	101766 non-null	int64
1	patient_nbr	101766 non-null	int64
2	race	101766 non-null	object
3	gender	101766 non-null	object
4	age	101766 non-null	object
5	weight	101766 non-null	object
6	admission_type_id	101766 non-null	int64
7	discharge_disposition_id	101766 non-null	int64
8	admission_source_id	101766 non-null	int64
9	time_in_hospital	101766 non-null	int64
10	payer_code	101766 non-null	object
11	medical_specialty	101766 non-null	object
12	num_lab_procedures	101766 non-null	int64
13	num_procedures	101766 non-null	int64
14	num_medications	101766 non-null	int64
15	number_outpatient	101766 non-null	int64
16	number_emergency	101766 non-null	int64
17	number_inpatient	101766 non-null	int64
18	diag_1	101766 non-null	object
19	diag_2	101766 non-null	object

race	int32
gender	int32
age	int32
admission_type_id	int64
discharge_disposition_id	int64
admission_source_id	int64
time_in_hospital	int64
num_lab_procedures	int64
num_procedures	int64
num_medications	int64
number_outpatient	int64
number_emergency	int64
number_inpatient	int64
diag_1	int32
diag_2	int32

	race	gender	age	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num_medications
1	2	0	1	0	0	6	2	58	0	17
2	0	0	2	0	0	6	1	10	5	12
3	2	1	3	0	0	6	1	43	1	15
4	2	1	4	0	0	6	0	50	0	7

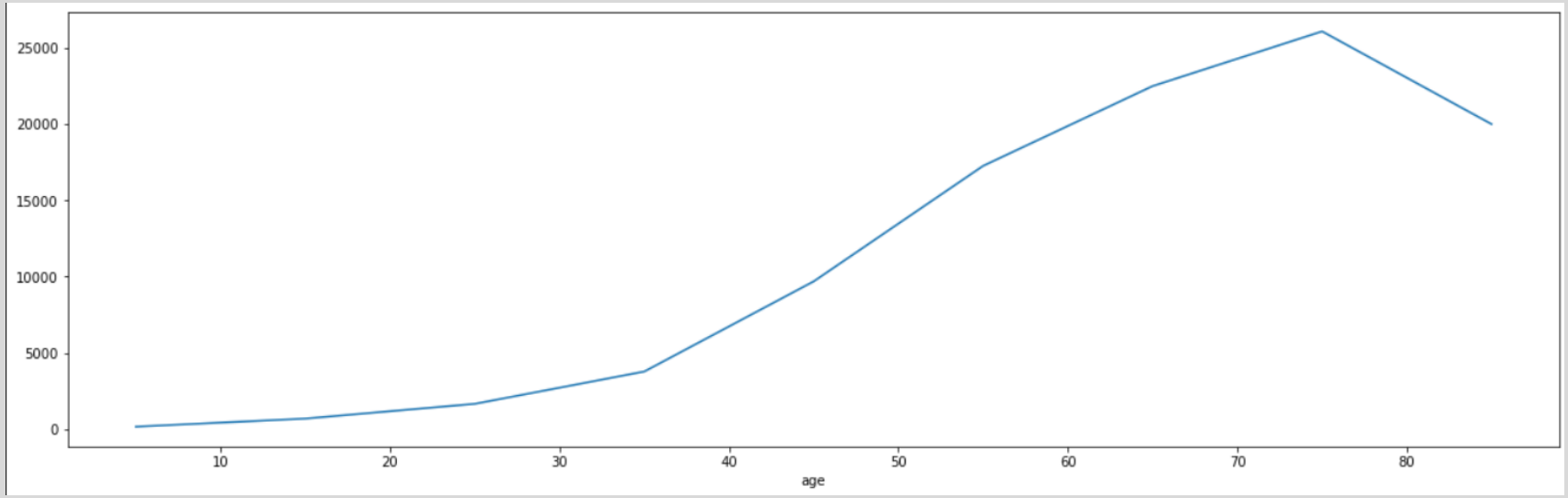
Data preprocessing:

The last step is the heatmap of the correlation matrix to find links between variables.



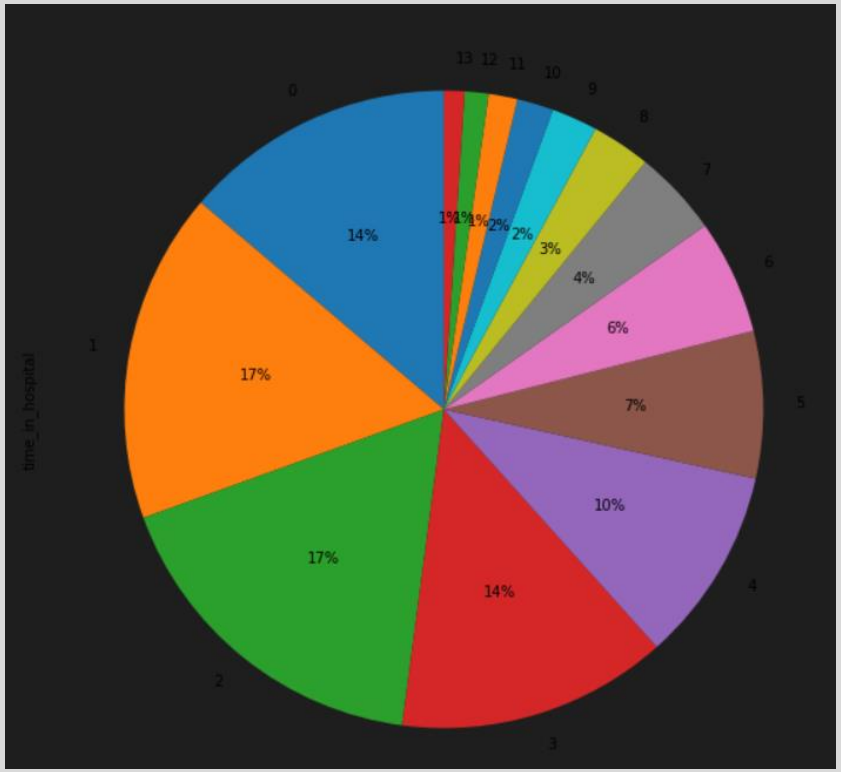
Data visualization:

First let's have a look to the number of diabetic people depending on their age:

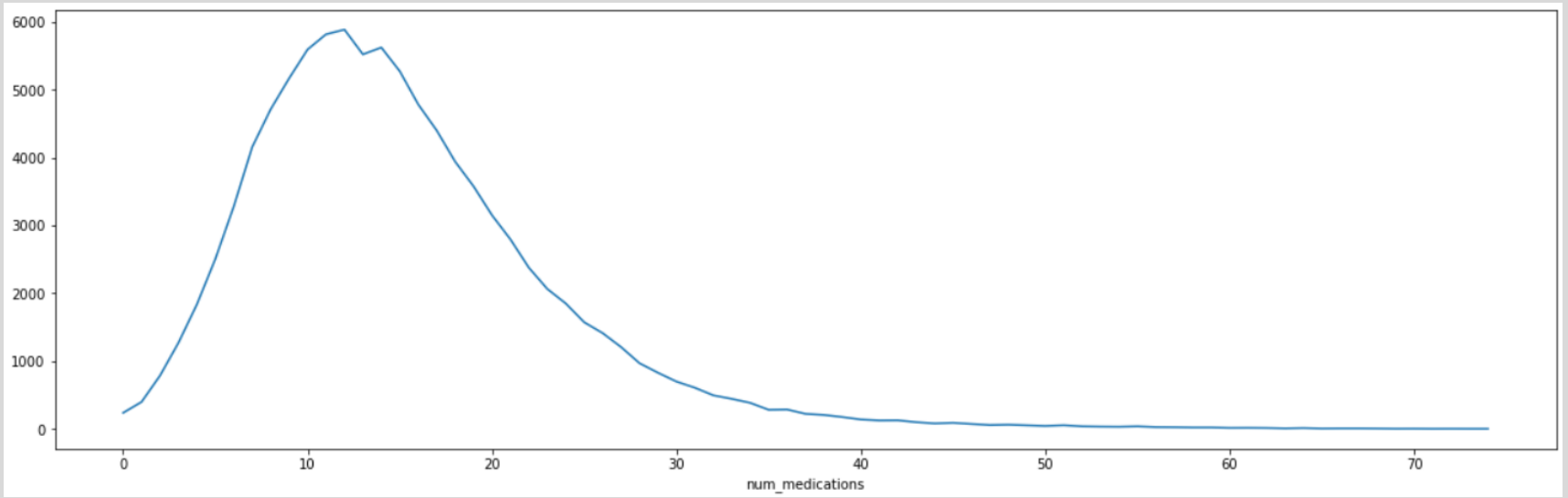


Data visualization:

Then, we have a repartition of patient time at the hospital (in days):

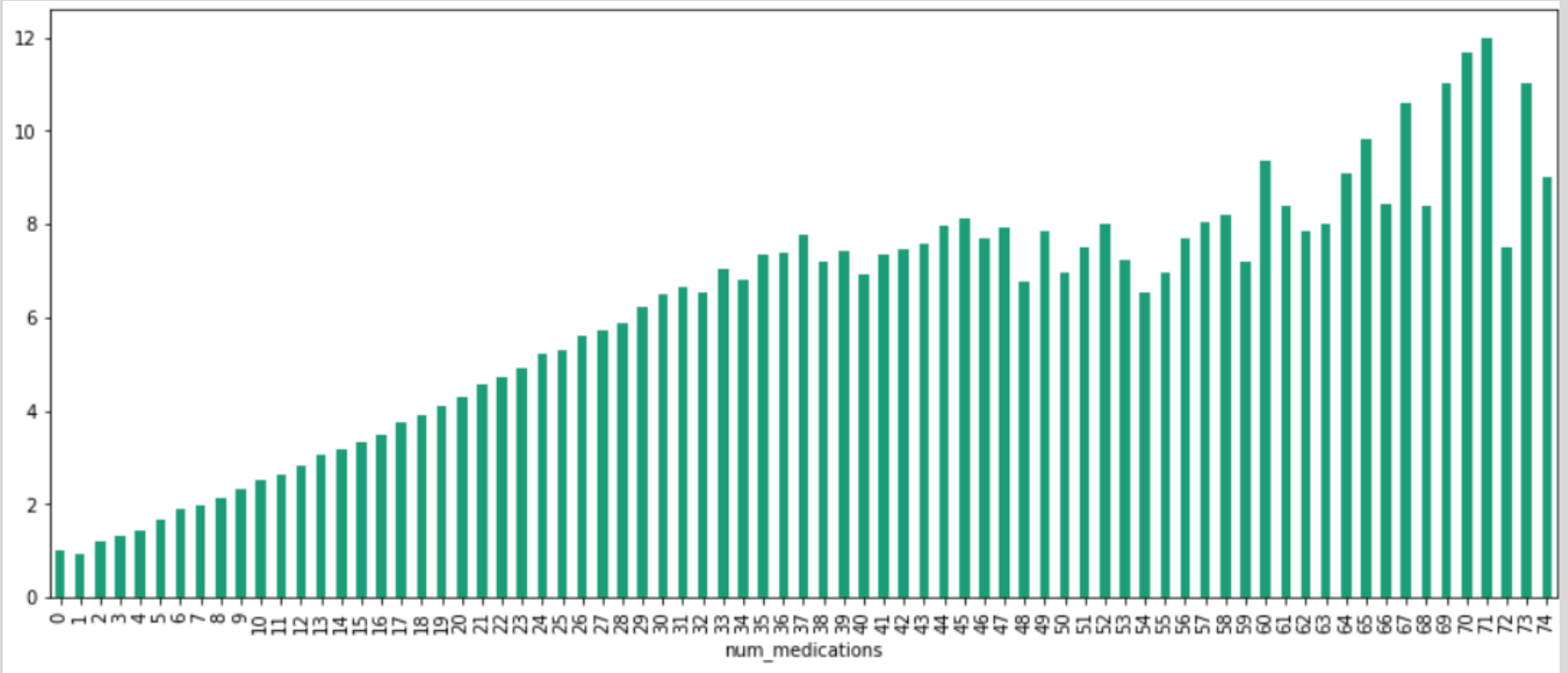


And a repartition of the medication they take during their stay:



Data visualization:

The last visualization is a representation of the time in hospital in function of the medication they take:



Modeling:

Metrics	Results
AUC	0.501
Accuracy	0.909
Recall	0.003
Precision	0.429
F1 score	0.006
Specificity	1.000
R^2	-0.101
Adjusted R^2	-0.103
Mean Absolute Error	0.091
Mean Squared Error	0.091
Rooted Mean Square Error	0.302

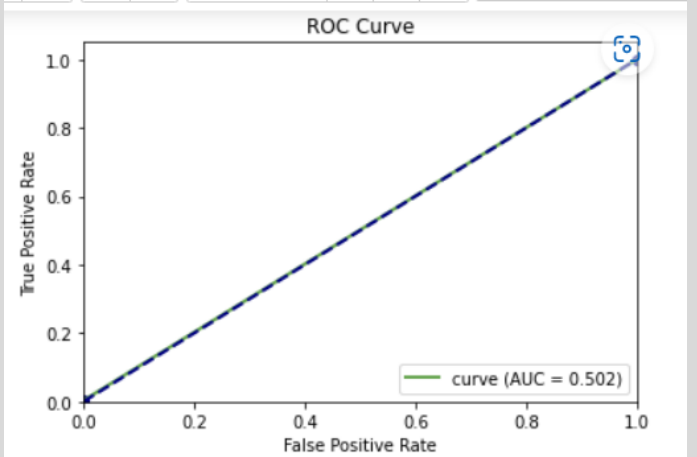
First results with a logistic regression

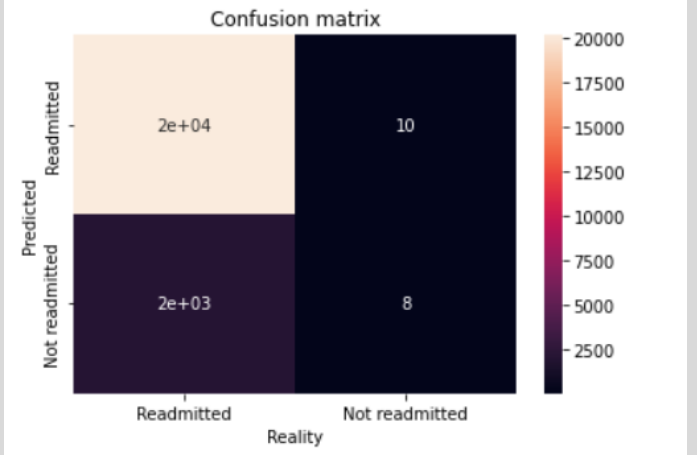
```
X, y = data.loc[:,data.columns!='readmitted'],data['readmitted']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(45091, 38) (22209, 38) (45091,) (22209,)
```

```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
lr = LogisticRegression()
lr.fit(X_train,y_train)
y_pred=lr.predict(X_test)
res_metrics(X_test,y_test,y_pred)
roc_auc( y_test, y_pred)
sns.heatmap(sklearn.metrics.confusion_matrix(y_test, y_pred),annot=True,
            xticklabels=["Readmitted", "Not readmitted"],
            yticklabels=["Readmitted", "Not readmitted"])
plt.ylabel("Predicted")
plt.xlabel("Reality")
plt.title("Confusion matrix")
plt.show()
```





Modeling:

Modification of dataset and Models Benchmark

```
sm = SMOTE(random_state=20)
train_input_new, train_output_new = sm.fit_resample(X_train, y_train)
print('New dataset shape {}'.format(Counter(train_output_new)))
```

```
Original dataset shape Counter({0: 41049, 1: 4042})
New dataset shape Counter({0: 41049, 1: 41049})
```

```
train_input_new = pd.DataFrame(train_input_new, columns = list(X.columns))
X_train, X_test, y_train, y_test = train_test_split(train_input_new, train_output_new, test_size=0.33)
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

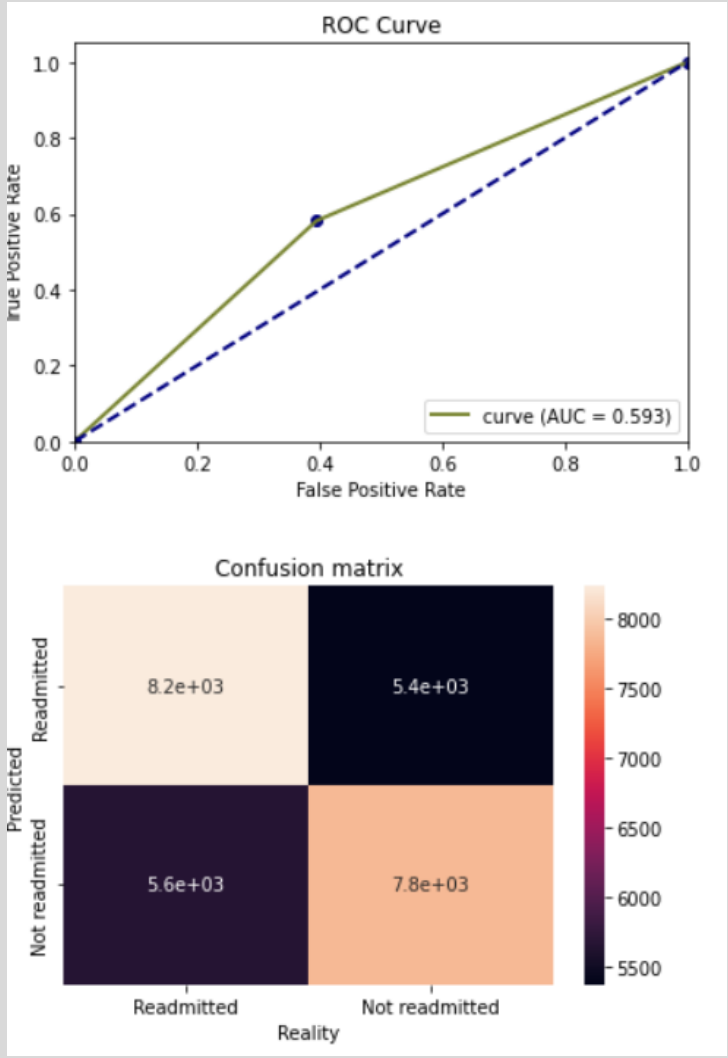
```
models = []
models.append(('LR', LogisticRegression()))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('NB', GaussianNB()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('RFC', RandomForestClassifier()))
models.append(('XGB', XGBClassifier()))
models.append(('ADA', AdaBoostClassifier()))
models.append(('MLPC', MLPClassifier(solver='lbfgs', alpha=1e5,hidden_layer_sizes=(5, 5), random_state=1)))
results=[]
names=[]
for name, model in models:
    kfold=StratifiedKFold(n_splits=10,shuffle=True,random_state=42)
    cv_results=cross_val_score(model,X_train,y_train,cv=kfold,scoring="accuracy")
    results.append(cv_results)
    names.append(name)
    msg="%s:%f" % (name,cv_results.mean())
    print(msg)
```

```
LR:0.584747
LDA:0.585365
NB:0.502700
KNN:0.800545
CART:0.893246
RFC:0.951659
XGB:0.950623
ADA:0.882629
MLPC:0.500882
```

Modeling:

Second results with a logistic regression

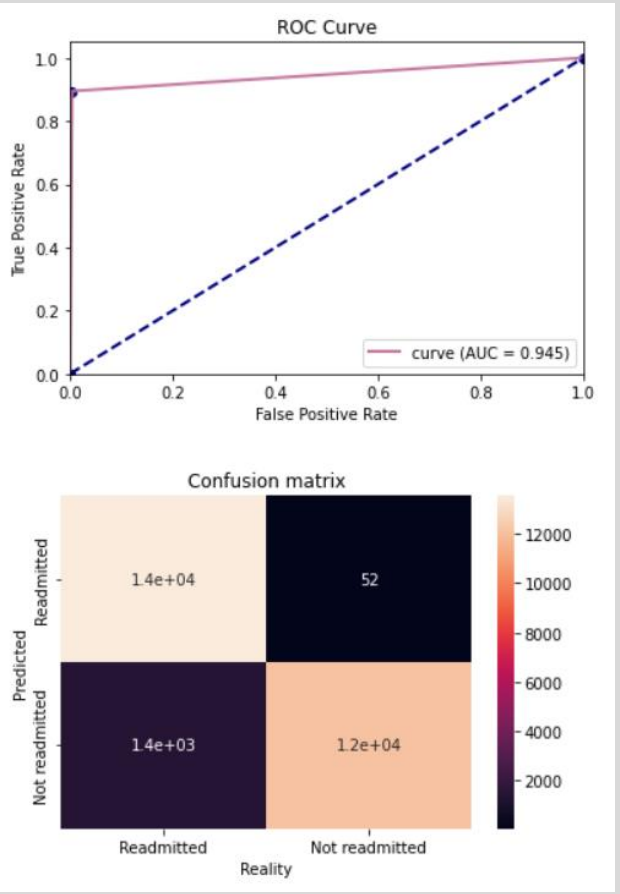
Metrics	Results
AUC	0.593
Accuracy	0.593
Recall	0.581
Precision	0.594
F1 score	0.588
Specificity	0.605
R^2	-0.626
Adjusted R^2	-0.629
Mean Absolute Error	0.407
Mean Squared Error	0.407
Rooted Mean Square Error	0.638



Modeling:

Results with Random Forest Classification

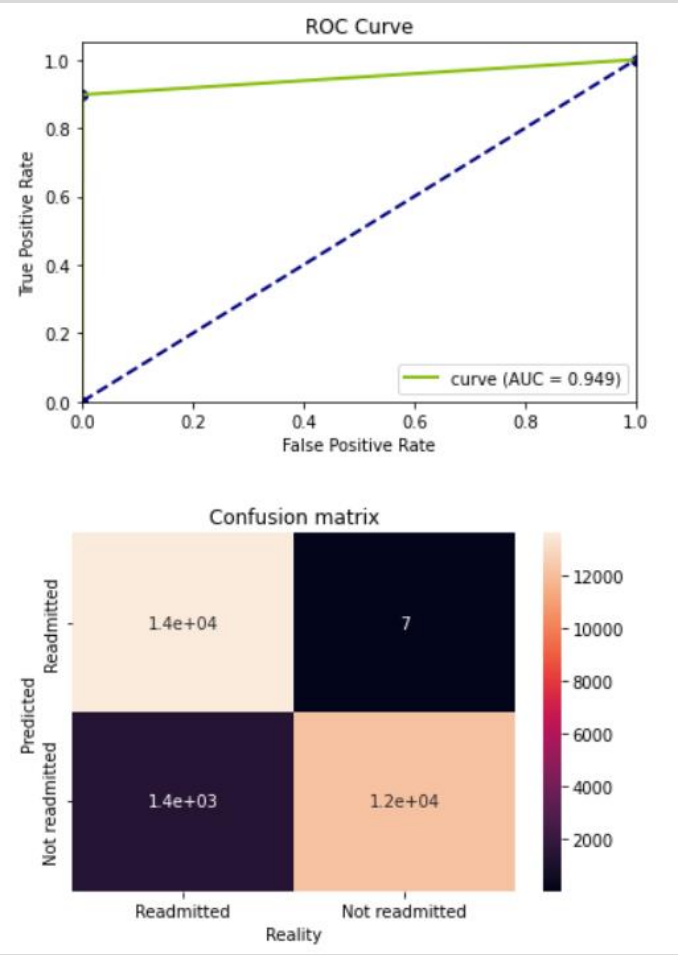
Metrics	Results
AUC	0.945
Accuracy	0.946
Recall	0.895
Precision	0.996
F1 score	0.942
Specificity	0.996
R^2	0.782
Adjusted R^2	0.782
Mean Absolute Error	0.054
Mean Squared Error	0.054
Rooted Mean Square Error	0.233



Modeling:

Results with XGBoost Classification

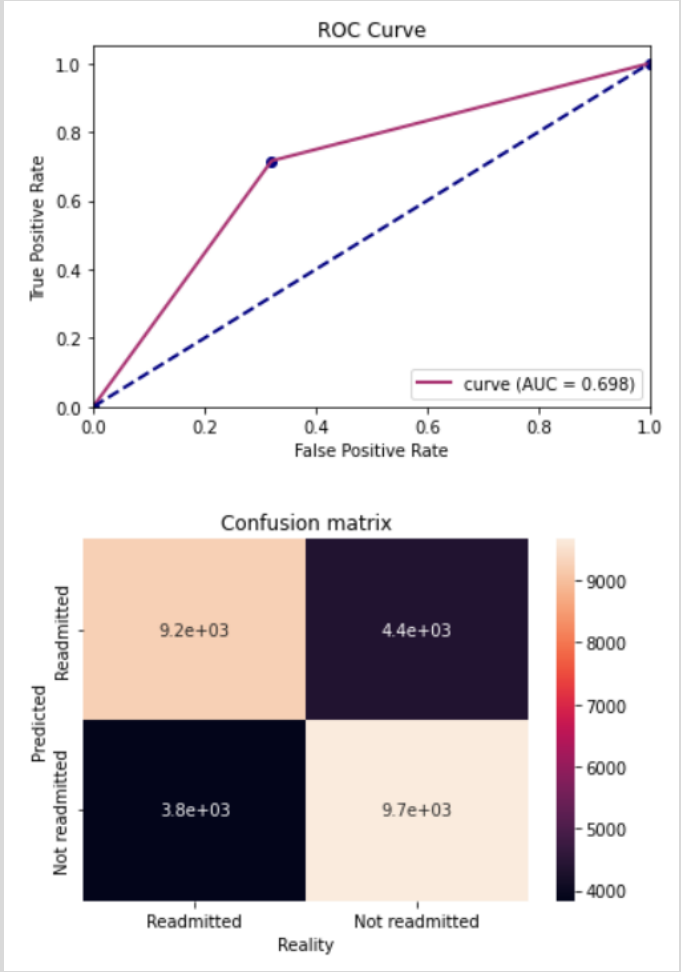
Metrics	Results
AUC	0.949
Accuracy	0.949
Recall	0.898
Precision	0.999
F1 score	0.946
Specificity	0.999
R^2	0.795
Adjusted R^2	0.795
Mean Absolute Error	0.051
Mean Squared Error	0.051
Rooted Mean Square Error	0.226



Modeling:

Results with Decision Tree Classification

Metrics	Results
AUC	0.698
Accuracy	0.698
Recall	0.716
Precision	0.689
F1 score	0.702
Specificity	0.680
R^2	-0.209
Adjusted R^2	-0.211
Mean Absolute Error	0.302
Mean Squared Error	0.302
Rooted Mean Square Error	0.550



Modeling:

Global results

Models	AUC	Accuracy	Recall	Precision	F1 score	Specificity	R^2	Adjusted R^2	Mean Absolute Error	Mean Squared Error	Rooted Mean Square Error
XGBoost	0.949	0.949	0.898	0.999	0.946	0.999	0.795	0.795	0.051	0.051	0.226
Random Forest	0.945	0.946	0.895	0.996	0.942	0.996	0.782	0.782	0.054	0.054	0.233
Decision Tree	0.698	0.698	0.716	0.689	0.702	0.680	-0.209	-0.211	0.302	0.302	0.550
Logistic Regression	0.593	0.593	0.581	0.594	0.588	0.605	-0.626	-0.629	0.407	0.407	0.638

Django API:

The databases

Diabetes Study

We chose to study the diabetes databases, here is our result on the project.

The dataframe contains 101766 rows and 50 columns

Here are the datas as we collected them.

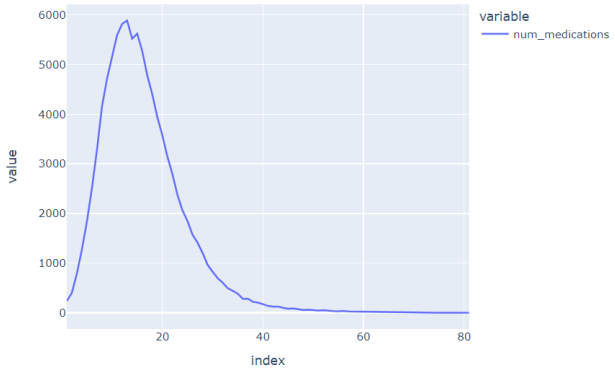
	race	gender	age	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	num_lab_procedures	num_procedures	num_medications	number_outpatient	number_e
1	Caucasian	Female	[10-20)	1	1	7	3	59	0	18	0	0
2	AfricanAmerican	Female	[20-30)	1	1	7	2	11	5	13	2	0
3	Caucasian	Male	[30-40)	1	1	7	2	44	1	16	0	0
4	Caucasian	Male	[40-50)	1	1	7	1	51	0	8	0	0
5	Caucasian	Male	[50-60)	2	1	2	3	31	6	16	0	0

And here are the datas after we cleaned them.

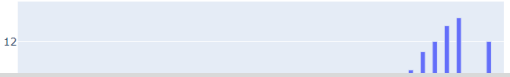
	encounter_id	patient_nbr	race	gender	age	weight	admission_type_id	discharge_disposition_id	admission_source_id	time_in_hospital	payer_code	medical_specialty	num_lab_procedures	nu
0	2278392	8222157	Caucasian	Female	[0-10)	?	6	25	1	1	?	Pediatrics-Endocrinology	41	0
1	149190	55629189	Caucasian	Female	[10-20)	?	1	1	7	3	?	?	59	0
2	64410	86047875	AfricanAmerican	Female	[20-30)	?	1	1	7	2	?	?	11	5
3	500364	82442376	Caucasian	Male	[30-40)	?	1	1	7	2	?	?	44	1
4	16680	42519267	Caucasian	Male	[40-50)	?	1	1	7	1	?	?	51	0

Some graphs

Here you can see the repartition of drugs and how much they were combined



Here is the time spend in the hospital depending on the number of medication that is taken



The project:

How does it fit in the context of our study?

Resume of all libraries use
In python during the semester

Use of models seen in machine learning class with a problematic

Observation of concept discovered
During soft skills sessions

A subject that could fit with
Problems of the medical sector

Conclusions:

Which lessons can we
Bring back from this study?

Models	AUC	Accuracy	Recall	Precision	F1 score	Specificity	R^2	Adjusted R^2	Mean Absolute Error	Mean Squared Error	Rooted Mean Square Error
XGBoost	0.949	0.949	0.898	0.999	0.946	0.999	0.795	0.795	0.051	0.051	0.226
Random Forest	0.945	0.946	0.895	0.996	0.942	0.996	0.782	0.782	0.054	0.054	0.233
Decision Tree	0.698	0.698	0.716	0.689	0.702	0.680	-0.209	-0.211	0.302	0.302	0.550
Logistic Regression	0.593	0.593	0.581	0.594	0.588	0.605	-0.626	-0.629	0.407	0.407	0.638

XGBoost allows us to predict the readmission of a patient depending on the drugs, age and admission with an accuracy of 94,9%.