ONCFM

Méthode d’identification des contrefaçons des billets en euros

Matthieu Gimbert - Juin 2022

Objectifs

Mise en place d'une modélisation capable d'identifier automatiquement les vrais des faux billets à partir simplement de certaines dimensions du billet et des éléments qui le composent.

1. Analyse du dataset
2. Gestion des valeurs manquantes
3. Modèle de classification binaire
4. Web app de vérification

I - Analyse du dataset

Fichier fourni pour l'apprentissage supervisé du modèle

```
df = pd.read_csv("billets.csv", sep=";")
df.head()
```

	is_genuine	diagonal	height_left	height_right	margin_low	margin_up	length
0	True	171.81	104.86	104.95	4.52	2.89	112.83
1	True	171.46	103.36	103.66	3.77	2.99	113.09
2	True	172.69	104.48	103.50	4.40	2.94	113.16
3	True	171.36	103.91	103.94	3.62	3.01	113.51
4	True	171.73	104.28	103.46	4.04	3.48	112.54

7 variables, 1500 individus

- 6 variables décimales représentant les différentes mesures du billet
- 1 variable booléenne indiquant l'authenticité du billet
- 500 faux billets, 1000 vrais billets
- 37 valeurs manquantes pour la variable margin_low

Pas d'outliers aberrants

Seules quelques valeurs atypiques utiles à conserver pour l'entraînement de notre modèle.

Distribution

Loi normale pour chacune des variables selon la variable is_genuine (Kurtosis élevé pour True et bas pour False)

Corrélation

Les variables length et margin_low sont fortement corrélées à is_genuine et négativement corrélées entre elles.

2 - Gestion des valeurs manquantes

La variable margin_low a 37 valeurs manquantes sur les 1500 individus, soit 2.5% du dataset total.

```
df.info()
```

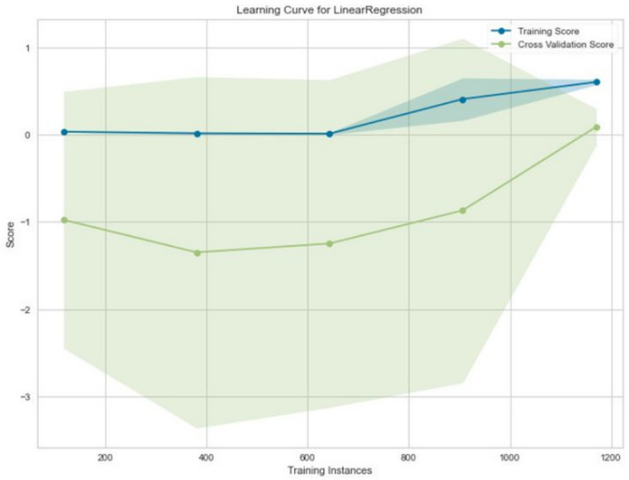
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1500 entries, 0 to 1499
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   is_genuine    1500 non-null   bool   
 1   diagonal      1500 non-null   float64
 2   height_left   1500 non-null   float64
 3   height_right  1500 non-null   float64
 4   margin_low    1463 non-null   float64
 5   margin_up     1500 non-null   float64
 6   length        1500 non-null   float64
dtypes: bool(1), float64(6)
memory usage: 71.9 KB
```

Plusieurs méthodes testées...

Interpolate method, polynomial regression, ridge and adaboost methods
Résultats médiocres (accuracy < 0.6)

une retenue (provisoirement)

Sans briller, la méthode de la régression linéaire donne les meilleurs résultats.
Selon la courbe d'apprentissage, il faudrait néanmoins un nombre d'individus plus grand pour améliorer le résultat (convergence).



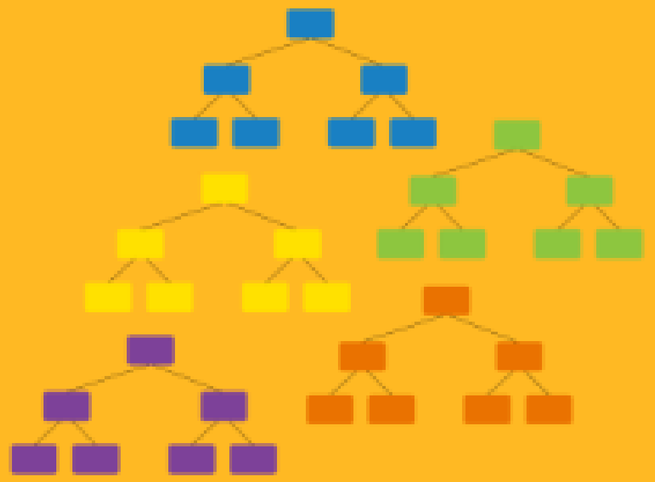
Stratégie

70% des valeurs manquantes concernent la classe dominante des vrais billets (1000 contre 500).
Suppression des valeurs manquantes

3 - Classification binaire

Random Forest

L'algorithme par forêt d'arbres décisionnels est un algorithme de classification qui effectue un apprentissage sur de multiples arbres de décision entraînés sur des sous-ensembles de données légèrement différents.



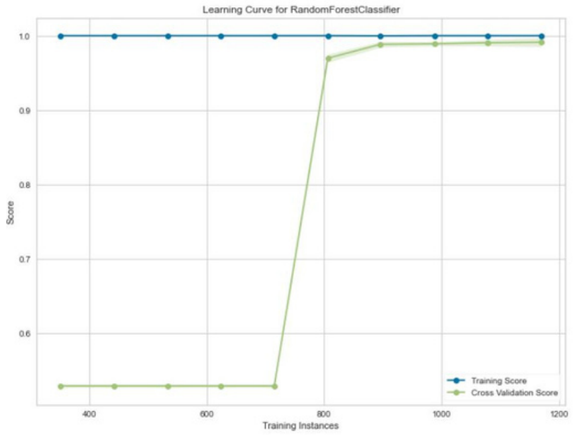
```
# Paramètres à retenir pour RandomForest
best_params = CV_rf.best_params_
best_params
```

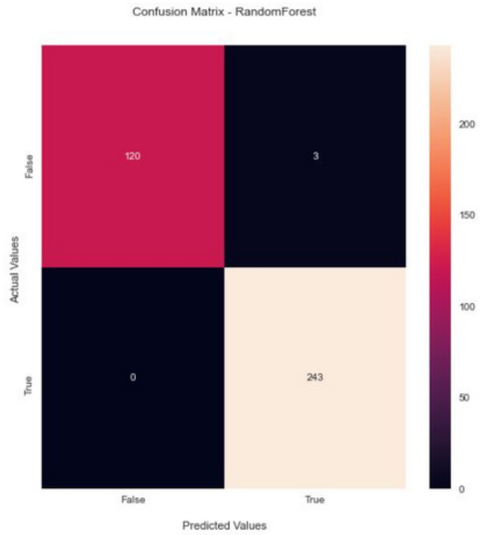
```
{'criterion': 'gini',
 'max_depth': 8,
 'max_features': 'auto',
 'n_estimators': 50}
```

```
rf = RandomForestClassifier(**best_params)
rf.fit(X_bin_train, y_bin_train)
y_bin_pred = rf.predict(X_bin_test)
```

```
print("Evaluation metrics - Random forest:")
print("Accuracy: " + str(metrics.accuracy_score(y_bin_test, y_bin_pred)))
print("F1 score: " + str(metrics.f1_score(y_bin_test, y_bin_pred, average="macro")))
```

```
Evaluation metrics - Random forest:
Accuracy: 0.9918032786885246
F1 score: 0.9907596758312505
```





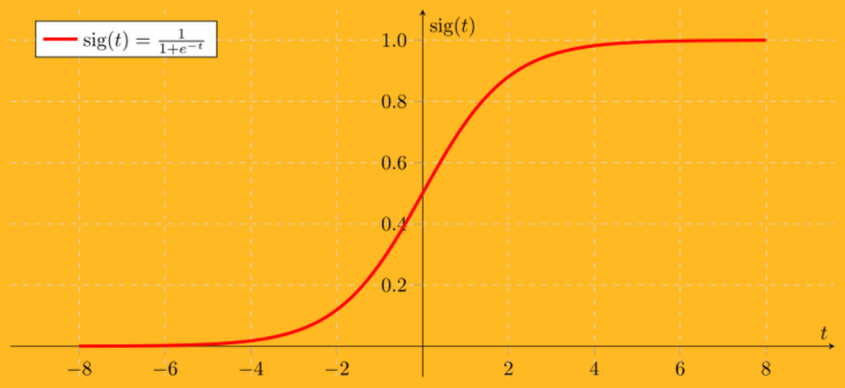
Résultat

Après hyper-paramétrage, le modèle Random Forest obtient une efficacité de détection de l'authenticité des billets légèrement supérieur à 99% des cas.
3 faux positifs sont observés.

3 - Classification binaire

Logistic Regression

La régression logistique est un algorithme supervisé de classification permettant d'évaluer et de caractériser les relations entre une variable réponse de type binaire et plusieurs variables explicatives.



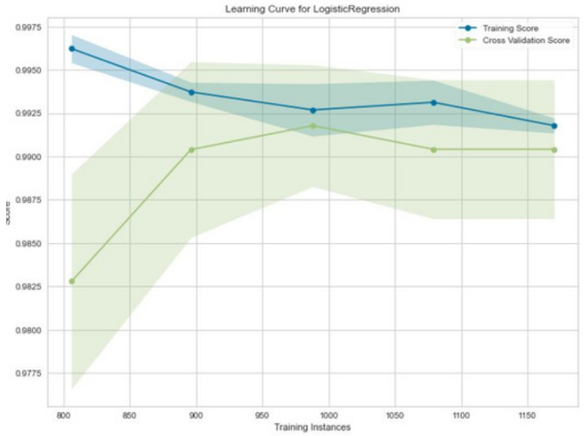

```
# Paramètres à retenir pour LogisticRegression
best_params_logr = CV_logr.best_params_
best_params_logr
```

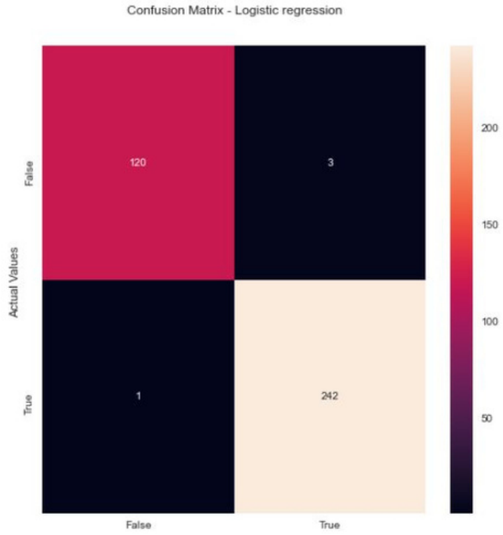
```
{'C': 10.0, 'penalty': 'l2', 'solver': 'newton-cg'}
```

```
logr = LogisticRegression(**best_params_logr)
logr.fit(X_bin_train, y_bin_train)
y_bin_pred_logr = logr.predict(X_bin_test)
```

```
print("Evaluation metrics - Logistic regression:")
print("Accuracy: " + str(metrics.accuracy_score(y_bin_test, y_bin_pred_logr)))
print("F1 score: " + str(metrics.f1_score(y_bin_test, y_bin_pred_logr, average
```

```
Evaluation metrics - Logistic regression:
Accuracy: 0.9890710382513661
F1 score: 0.9877049180327868
```





Résultat

Après hyper-paramétrage, le modèle Logistic Regression obtient une efficacité de détection de l'authenticité des billets légèrement inférieur à 99% des cas.
3 faux positifs et 1 faux négatifs sont observés.

3 - Classification binaire

K-means

Bien que dédié à l'apprentissage non supervisé, testons cette méthode de partitionnement de données (en supprimant la target de notre dataset - is_genuine).


```
df_kmeans['Diff'].value_counts()
```

```
1    1481
0      19
Name: Diff, dtype: int64
```

```
labels = kmeans.labels_

# check how many of the samples were correctly labeled
correct_labels = sum(y_km == labels)

print(f"Result: {correct_labels} out of {y_km.size} samples were correctly labeled.")
```

```
Result: 1481 out of 1500 samples were correctly labeled.
```

```
print(f"Accuracy score: {correct_labels/float(y_km.size):.2f}")
```

```
Accuracy score: 0.99
```

Résultat

La méthode k-means obtient une efficacité de détection de l'authenticité des billets dans 99% des cas.

3 - Classification binaire

Conclusion

Autres modèles

2 autres méthodes ont été testées, Multi Layer Perceptron (hidden_layer_sizes=[100]) et Support Vector Machine (C=10, gamma=0.1), sans donner de résultats plus performants.

K-means

Le modèle K-means est très pertinent puisque très économe en temps (il n'est pas nécessaire de faire la classification manuelle du dataset d'entraînement) mais plus "risqué".

Logistic regression

Le modèle de régression logistique est plus rapide d'un facteur 10 que le modèle Random Forest mais "moins" performant.

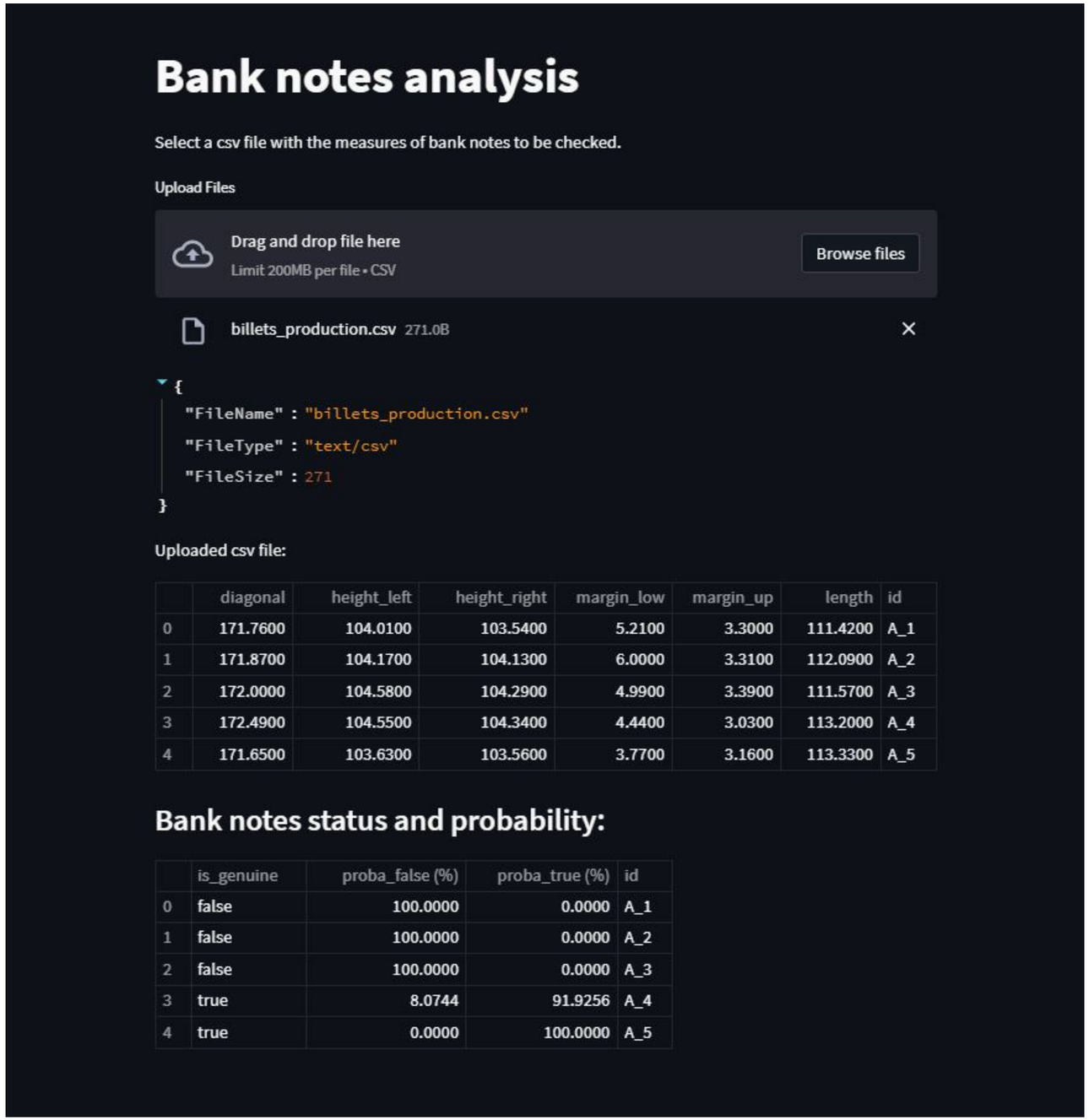
Modèle retenu : Random Forest

4 - Web app

Fichier de vérification

Utilisation de la librairie Streamlit pour la création d'une web app permettant la vérification de l'authenticité des billets de banque (batch)

Streamlit



	diagonal	height_left	height_right	margin_low	margin_up	length	id
0	171.7600	104.0100	103.5400	5.2100	3.3000	111.4200	A_1
1	171.8700	104.1700	104.1300	6.0000	3.3100	112.0900	A_2
2	172.0000	104.5800	104.2900	4.9900	3.3900	111.5700	A_3
3	172.4900	104.5500	104.3400	4.4400	3.0300	113.2000	A_4
4	171.6500	103.6300	103.5600	3.7700	3.1600	113.3300	A_5

	is_genuine	proba_false (%)	proba_true (%)	id
0	false	100.0000	0.0000	A_1
1	false	100.0000	0.0000	A_2
2	false	100.0000	0.0000	A_3
3	true	8.0744	91.9256	A_4
4	true	0.0000	100.0000	A_5