Python for Data Analysis
Final Project

Thibault CHASSEFAIRE, Emmanuel MENEGHETTI

ESILV
ENGINEERING SCHOOL
DE VINCI PARIS

Dataset

- Nous avons un dataset contenant les données de prévision de la faillite d'entreprises polonaises. Les entreprises en faillite ont été analysées au cours de la période 2000-2012, tandis que les entreprises encore en activité ont été évaluées de 2007 à 2013.

- Nous nous demandons quelles sont les colonnes qui sont corrélées entre elles.

Méthodologie – 1e méthode ACP

- Nous commençons par mettre les colonnes en relation les unes avec les autres en faisant une recherche automatique de corrélation entre les colonnes
- Nous regardons ensuite les différentes corrélations pour voir quelles colonnes sont plus ou moins liées
- Nous ne gardons pour l'analyse que les corrélations élevées. Pour ce faire, nous faisons une Analyse en Composantes Principales (ACP)

Normaliser les données

- On normalise les données pour les « centrer-réduire » et pouvoir ensuite faire dessus des manipulations mathématiques.
  En effet, toutes les variables doivent être à la même échelle pour appliquer l'ACP. Nous utilisons StandardScaler de scikit-learn pour normaliser les caractéristiques de l'ensemble de données sur l'échelle des unités (moyenne = 0 et variance = 1).

Normaliser les données

Entrée [4]:
```
print(df.shape)
df = df.dropna()
df.shape
```

```
(7027, 65)
```

Out[4]: (3194, 65)

Entrée [5]:
```
    features = []
    for i in range(1,65):
        features.append("Attr"+str(i))
    features.append("class")
    x = df.loc[:, features].values
    x = StandardScaler().fit_transform(x)
    pd.DataFrame(data = x, columns = features).head()
```

Out[5]:

Attr10	...	Attr56	Attr57	Attr58	Attr59	Attr60	Attr61	Attr62	Attr63
.275097	...	0.303058	0.355870	-0.317356	-0.063455	-0.156390	-0.173784	-0.098988	-0.233386
.292913	...	0.754929	-0.262617	-0.597844	-0.002817	-0.142681	-0.204375	0.179430	-0.408496
.876950	...	0.229546	0.377561	-0.244440	-0.050629	-0.146281	-0.157022	-0.465006	0.469223
.503561	...	0.943992	-0.162990	-0.953098	-0.028708	-0.158192	-0.253596	-0.283601	-0.008766
.451886	...	-0.461549	-0.364705	0.441094	-0.031491	-0.200169	0.037214	-0.517338	0.730235

Sélectionner les composantes principales

- Nous sélectionnons les composantes principales les plus significatives avec la commande explained_variance. Nous sélectionnons celles qui représentent 95% de la variance, soit, après tâtonnement, les 29 colonnes les plus significatives sur les 64 au départ.

Sélectionner les composantes principales

Entrée [7]:

```
explained_variance = pca.explained_variance_ratio_
compteur = 0
for i in range(0, 29):
    compteur += explained_variance[i]
print(compteur)
```

```
0.9439891563081817
```

Entrée [8]:

```
pca = PCA(n_components=29)
principalComponents = pca.fit_transform(x)
for i in range(0, 29):
    df['PC' + str(i + 1)] = principalComponents[:, i]
df.head()
# We reduced number of axis from 64 to 30 (~45% of initial data was kept)
# Together, the first 30 principal components contain 95% of the information
```

Out[8]:

	Attr1	Attr2	Attr3	Attr4	Attr5	Attr6	Attr7	Attr8	Attr9	Attr1
0	0.200550	0.37951	0.396410	2.0472	32.351	0.388250	0.249760	1.33050	1.1389	0.5049
8	0.009020	0.63202	0.053735	1.1263	-37.842	0.000000	0.014434	0.58223	1.3332	0.3679
12	0.266690	0.34994	0.611470	3.0243	43.087	0.559830	0.332070	1.85770	1.1268	0.6500
13	0.067731	0.19885	0.081562	2.9576	90.606	0.212650	0.078063	4.02900	1.2570	0.8011
14	-0.029182	0.21131	0.452640	7.5746	57.844	0.010387	-0.034653	3.73240	1.0241	0.7886

5 rows × 94 columns

Utilité de l'ACP

- Grâce à l'ACP, en gardant 95% des valeurs et en passant de 64 axes à 30, on a compressé 55% du dataset en ne perdant que 5% des valeurs.
  Cela permet donc d'éliminer du volume pour une classification future.

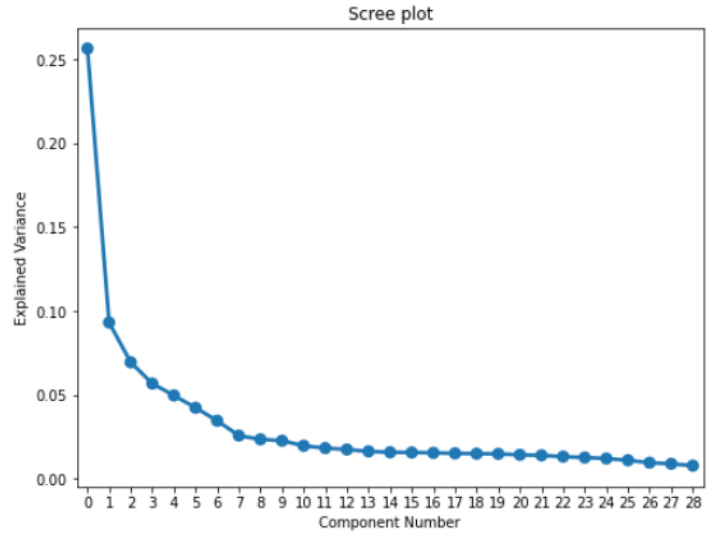
Corrélations grâce à l’ACP

- Nous mettons sous forme d’un diagramme de points le pourcentage de corrélation de chaque axe. Nous pouvons voir qu’à lui seul le premier axe représente plus de 25% de corrélation avec les différentes variables.

Corrélations grâce à l'ACP

Entrée [10]:

```
ind = np.arange(0, 29)
(fig, ax) = plt.subplots(figsize=(8, 6))
sns.pointplot(x=ind, y=pca.explained_variance_ratio_)
ax.set_title('Scree plot')
ax.set_xticks(ind)
ax.set_xticklabels(ind)
ax.set_xlabel('Component Number')
ax.set_ylabel('Explained Variance')
plt.show()
```



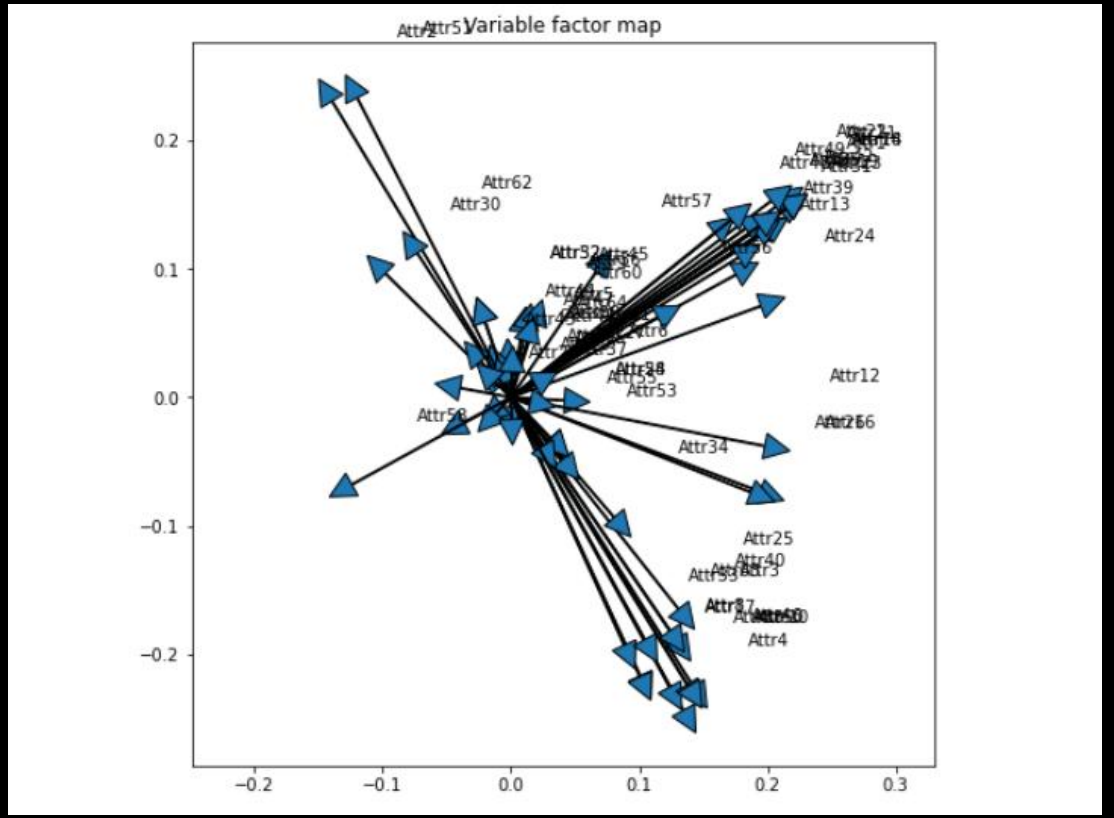
Cercle de corrélation

- A partir de l'ACP, nous allons essayer une analyse avec un cercle de corrélation, pour voir quelles données sont plus ou moins corrélées entre elles, avec les méthodes.
- Nous nous retrouvons avec un cercle de corrélation contenant 64 flèches. On voit que certaines flèches vont dans la même direction et donc que les données sont plus ou moins corrélées entre elles.

Cercle de corrélation

- Pour mieux voir, nous avons zoomé sur le cercle en appliquant un facteur 1000.
- En zoomant, nous nous rendons compte que de nombreuses variables sont corrélées, ce qui explique les 95% avec seulement la moitié des axes.

Cercle de corrélation



Résultats

- Au final, nous voyons que la colonne 24 et la 13 sont très corrélées entre elles. C’est assez logique. En effet, la colonne 24 représente *(marge brute (en 3 ans) / total des actifs)* et la 13 *((marge brute + amortissement) / ventes)*
- De même pour les colonnes 62 et 30, qui représentent respectivement *((dettes à court terme *365) / ventes) et ((total du passif - espèces) / ventes)*, les dettes faisant partie du passif, cela semble logique aussi.
- Nous ne détaillerons pas toutes les corrélations ici, celles-ci étant nombreuses.

Prédiction

- Nous revenons aux données brutes pour faire un modèle en deep learning qui prédira la faillite des entreprises. La colonne ‘class’, contient un 0 si l’entreprise n’a pas fait faillite et un 1 si elle a fait faillite.
- Nous faisons un réseau de neurones nous donnant une prédiction avec une précision d’environ 99% pour une perte d’environ 0,08 (binary crossentropy)

Prédiction

Entrée [9]:

```
model = mod.Sequential()

model.add(lyrs.Dense(500, input_shape = (64,)))
model.add(lyrs.Activation('relu'))
model.add(lyrs.Dropout(0.25))

model.add(lyrs.Dense(250))
model.add(lyrs.Activation('relu'))

model.add(lyrs.Dense(1))
model.add(lyrs.Activation('sigmoid'))
model.compile(optimizer="adam", loss='binary_crossentropy', metrics=['accuracy'])

history_min = model.fit(train,
          train_target,
          epochs=10,
          batch_size = 10,
          verbose=1,
          validation_data=(test, test_target))
score = model.evaluate(train, train_target, batch_size=10)
print(score)
```

Prédiction

```
Epoch 8/10
1598/1598 [==============================] - 2s 1ms/step - loss: 0.1794 - accuracy: 0.9772 - val_loss: 0.1090 - val_accurac
y: 0.9805
Epoch 9/10
1598/1598 [==============================] - 2s 1ms/step - loss: 0.2037 - accuracy: 0.9768 - val_loss: 0.1051 - val_accurac
y: 0.9805
Epoch 10/10
1598/1598 [==============================] - 2s 1ms/step - loss: 0.1620 - accuracy: 0.9773 - val_loss: 0.2186 - val_accurac
y: 0.9805
1598/1598 [==============================] - 1s 508us/step - loss: 0.2542 - accuracy: 0.9778
[0.25417032837867737, 0.977774977684021]
```

API

API


ACCUEIL
DESCRIPTION
CORRÉLATIONS
CLASSIFICATION
Analyse de dataset
Description du DataSet
Retrouvez les informations relatives au jeu de données
Correlations
Réduction du nombre de colonnes via une ACP
Classification
Obtenez une classification des résultats via un modèle de deep learning

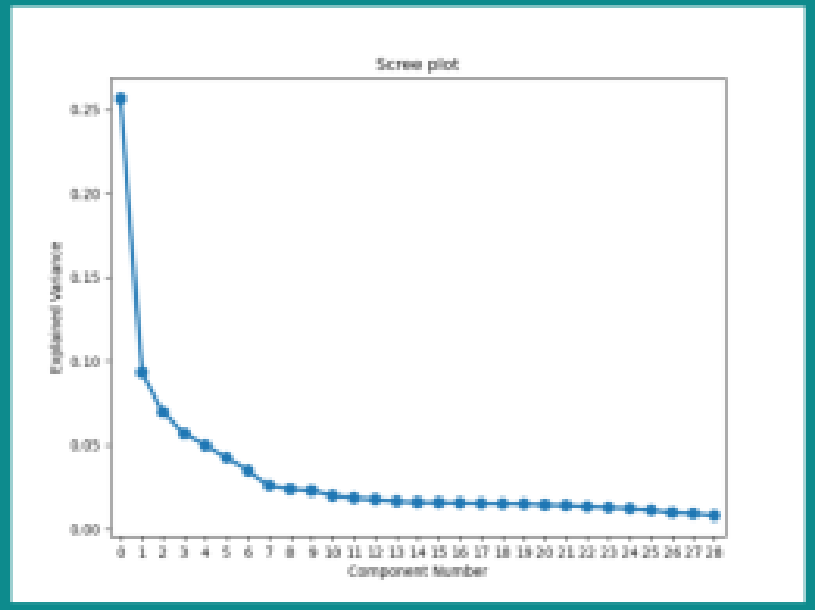

ACCUEIL
DESCRIPTION
CORRÉLATIONS
CLASSIFICATION
Description du jeu de données
The dataset is about bankruptcy prediction of Polish companies. The data was collected from Emerging Markets Information Service (EMIS), which is a database containing information on emerging markets around the world. The bankrupt companies were analyzed in the period 2000-2012, while the still operating companies were evaluated from 2007 to 2013.

API

ACCUEIL DESCRIPTION CORRÉLATIONS CLASSIFICATION

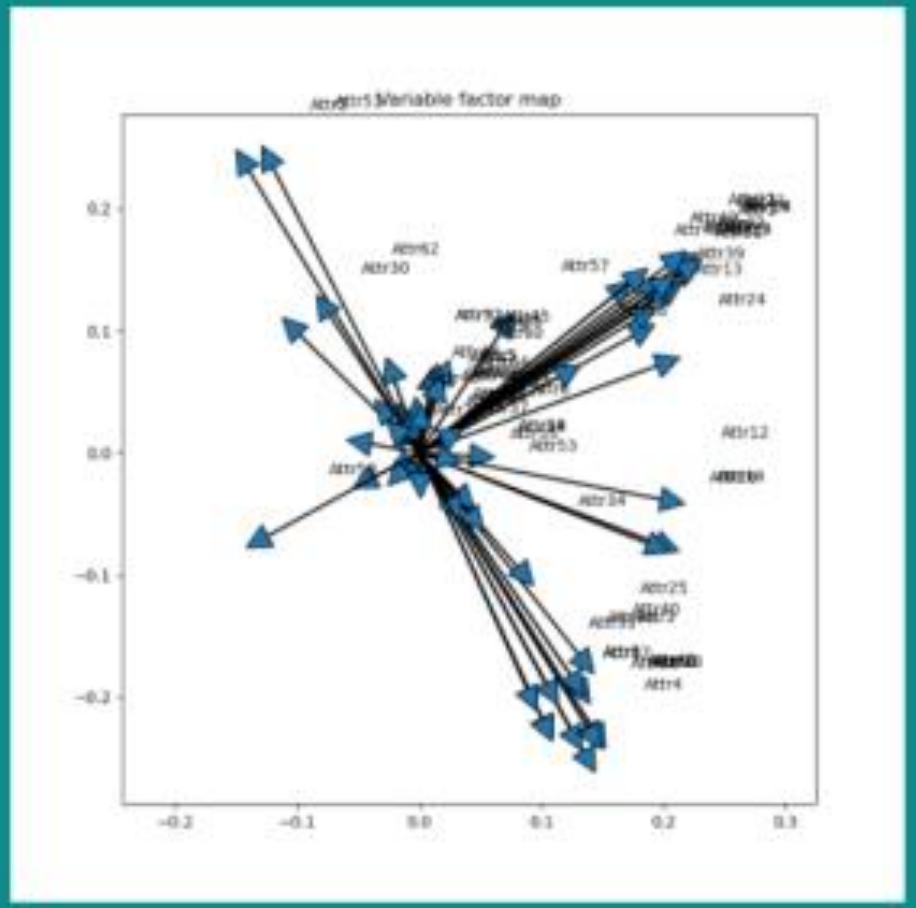
Correlations du dataset

Nous avons fait appel à une Analyse de Composante Principale (ACP) pour étudier étudier les éventuelles réductions de dataset. Les variables sont très corrélées entre-elles. En effet, 95% de l'information est contenue dans moins de 50% du jeu de données. Ce qui permet d'alléger énormément le dataset pour des manipulations futures. Ces images sont recalculées à chaque rafraîchissement de page. Elles ne sont pas inscrites en dur.



Participation à la variance de chaque composante

API



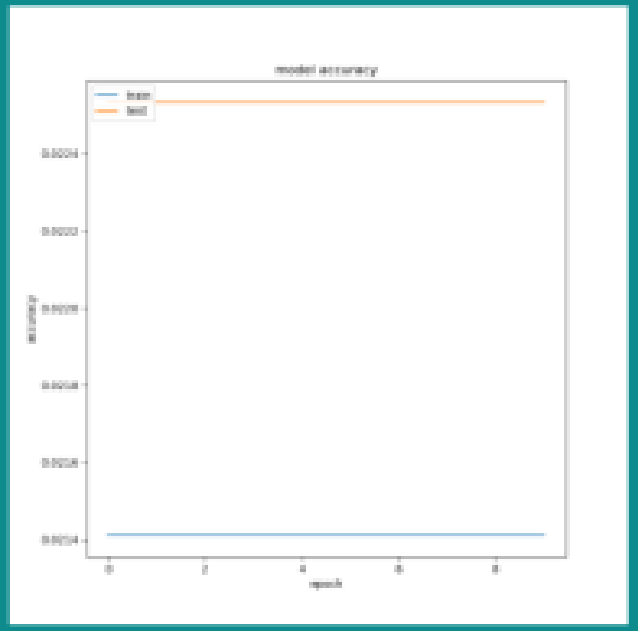
Corrélation de chaque composantes initiales entre-elles

API

ACCUEIL DESCRIPTION CORRÉLATIONS CLASSIFICATION

Classification using deep-learning

Nous avons fait appel à un réseau de neuronne permettant de prédire si une entreprise va faire faillite à partir de son bilan annuel. Nous avons construit deux modèles. Un avec une activation softmax et le second avec l'activation sigmoid (qui donne de meilleurs résultats). Les modèles sont recompilés à chaque chargement de page (d'où la lenteur pour générer cette page). Vous n'aurez jamais deux fois les même schémas. Ceux-cis représentent la précision du modèle et les pertes en fonction du nombre d'epoch sur l'échantillon d'entraînement et de test. Nous avons pris toutes les précautions nécessaires pour éviter tout sur-apprentissage.



Accuracy du premier modèle avec activation softmax