Machine Learning

Metz Numeric School

M2I

Florian AUBRY - 2024

Programme

- **Jour 1:**
  - Fondamentaux du Machine Learning
  - Pratique: Modèles supervisés
- **Jour 2**
  - Modèles avancés et optimisation
  - Pratique: Introduction aux modèles non supervisés

Programme

- **Jour 3**
  - Introduction au Deep Learning
  - Pratique: Application à l’analyse d’image et Traitement des langues
- **Jour 4**
  - Analyses avancées de données
  - Pratique: Combinaison de modèles en cas complexe

Sommaire

1. Analyse de composantes
2. Méthodes de réduction de dimension

Analyse de composantes

- Objectifs:
  - Réduire le coût de l'entraînement
  - Possiblement augmenter la performance du modèle
- Deux approches:
  - Non supervisée: sans la target
  - Supervisée: avec la target
- Trois catégories:
  - Wrapper: Plusieurs modèles, différentes features
  - Filtres: Analyse statistique de relation avec la target
  - ML: Fonction intrinsèque au modèle (RandomForest, ...)
- /!\ Différent de la réduction de dimensions qui combine les features

Analyse de composantes

Algorithme de **FILTRE** en fonction des catégories de données:

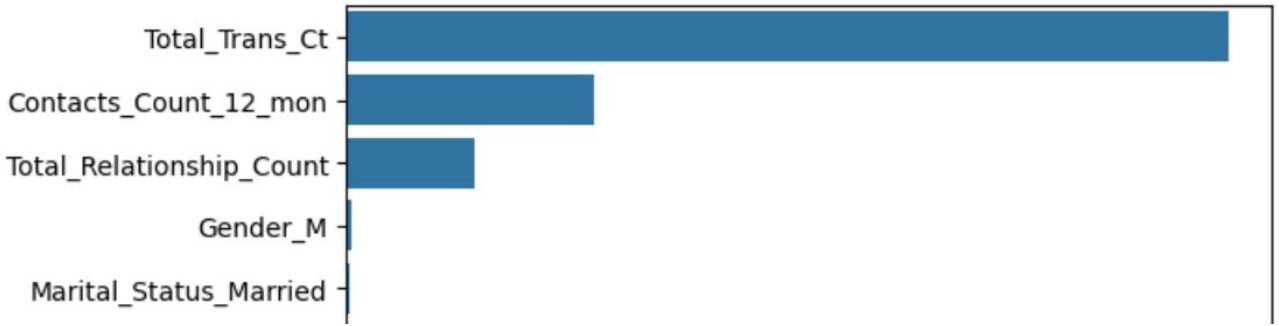
- Entrée numérique / Sortie numérique
  - Corrélation de Pearson (L) ou Spearman (NL)
- Entrée numérique / Sortie catégorie (et inversement)
  - Corrélation ANOVA (L)
  - Corrélation de Kendall (NL)
- Entrée catégorie / Sortie catégorie
  - Test Chi 2
  - Mututal Information Score => Méthode la plus agnostique sur l'ensemble des catégories en pratique

*NL: Non linéaire / L : Linéaire

Analyse de composantes

```python
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
kbest = SelectKBest(score_func=f_classif, k='all')
kbest.fit(x_train, y_train)

kbest_scores = pd.DataFrame({'Feature':x_train.columns, 'Score':kbest.scores_})
kbest_scores.sort_values(by='Score', ascending=False, axis=0, inplace=True)
kbest_scores.reset_index(drop=True, inplace=True)
```



Analyse de composantes

Algorithme de **WRAPPER**:

- Forward/Backward Feature Selection:
  - Ajout/Suppresion des features une par une
- Exhaustive Feature Selection:
  - Bruteforce des combinaisons possibles
- Recursive Feature Elimination
  - Modèle associant des poids aux features pour obtenir le meilleur modèle

Analyse de composantes

```
from sklearn.feature_selection import RFE
from sklearn.svm import SVR

model = SVR(kernel="linear")
selector = RFE(model, n_features_to_select=5, step=1)
selector = selector.fit(X_train, y_train)

selector.support_
# array([ True,  True,  True,  True,  True, False, False, False, False,
       False])
selected_features = X.columns[rfe.support_]
selector.ranking_
# array([1, 1, 1, 1, 1, 6, 4, 3, 2, 5])
```

Analyse de composantes

- Choix de la méthode d’analyse:
  - Il faut tester
  - Dépend de la nature, de la répartition etc des données

Réduction de dimensions

Réduction de dimensions:

- Principal Component Analysis (PCA):
  - Transforme les données en composants linéaires non corrélés entre eux
- Linear Discriminant Analysis (LDA):
  - Similaire au PCA avec un focus sur la séparation entre les classes (à l'origine fait pour de la classification)
- t-Distributed Stochastic Neighbor Embedding (t-SNE):
  - Conçu pour identifier des clusters entre les points (avec la similarité) et représenter les données en 2D ou 3D

Réduction de dimensions

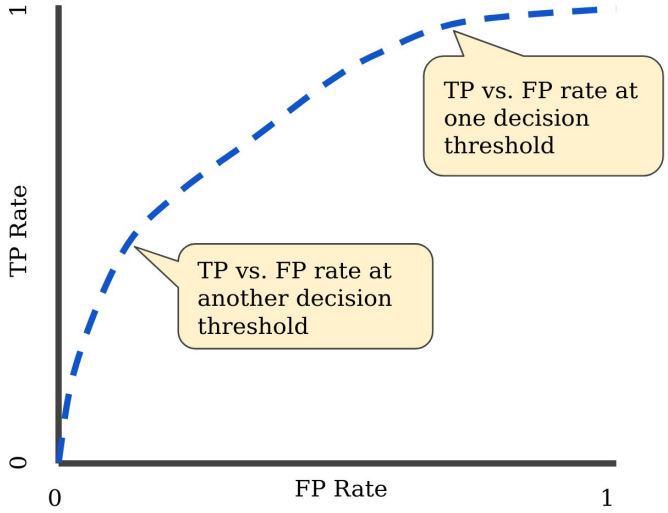
Réduction de dimensions:

- Il existe d’autres solutions:
  - Autoencodeurs
  - Independent Component Analysis (ICA)
  - Nonnegative Matrix Factorization
  - ...

Réduction de dimensions

- Encore une fois, le choix se fait de manière exploratoire
- L’analyse de feature n’exclue pas la réduction de dimension
  - Il est possible d’injecter les features les plus importantes avec les colonnes issues d’une réduction de dimension pour ajouter de l’information

Métriques avancées



Métrique ROC: Receiver Operating Characteristic

- Défini le ratio entre le taux de Faux Positifs et le taux de Vrais Positifs

Métrique AUC: Area Under the Curve

- Surface sous la courbe ROC
- Interprétation:
  - Proche de 1.0: classification parfaite
  - Proche de 0.5: incapacité à classifier

Métriques avancées

```
rnn_model.compile(optimizer=tf.keras.optimizers.Adam(1e-3),
                  loss=keras.losses.CategoricalCrossentropy(from_logits=True),
                  metrics=[tf.metrics.AUC()])
```

Partie Pratique

Vous disposez d'un jeu de données contenant des articles classés suivant le degré de désinformation (fake news)

- Réalisez une étude préalable des données (nombre de mots, longueur des phrases, équilibrage des classes, ...)
- Effectuez une première phase de nettoyage manuelle des données ainsi que de normalisation (à vous de chercher différentes solutions). On peut partir du principe que les mots utilisés sont plus importants que la syntaxe, les fake news cherchant à être plus choquante/percutantes
- Si besoin, ajoutez de nouvelles features (sentiment, ...)
- Réduisez le nombre de classes dans un premier temps si nécessaire

Partie Pratique

- En utilisant la librairie de votre choix, évaluez différents modèles et architectures pour trouver le modèle le plus adapté
- Vous devrez sauvegarder votre modèle et le fournir avec votre analyse (attention à tout inclure dans le modèle, y compris la vectorisation !)
- Si besoin, vous pouvez commencer par travailler avec seulement deux classes et réduire la quantité de données

Partie Pratique

- Vous devrez fournir un notebook commenté détaillant votre analyse et vos résultats, comme si vous vendiez cette analyse à un client qui ne connaît pas le Machine Learning
- Vous devrez expliquer en termes simples mais avec une justification vos choix et observations