



EuroVision Winner Prediction

UE : IRT 4

réalisé par :

Farah ABDELLI



Table des matières

Liste des Figures.....	4
Liste des abréviations.....	5
Introduction générale	6
Chapitre 1 : Compréhension du métier.....	7
I. Business objective	8
II. Assess the situation	8
III. Data mining goals.....	7
IV. Plan du projet	8
1. Étapes de planification du projet	8
2. Outils et technologies utilisés	9
Chapitre 2 : Compréhension des données	12
I. Collecte de données	12
1. Contestants	12
2. Votes	13
3. Scraping des années manquantes.....	14
4. Collecte EuroVision 2016-2022	15
II. Description des données.....	17
1. Contestants	17
2. Votes	18
3. Collecte EuroVision 2016-2022	18
III. Exploration des données.....	19
1. Contestants	19
2. Votes	20
3. Collecte EuroVision 2016-2022	21
IV. Vérification de la qualité des données	22
1. Contestants	22
2. Votes	23
3. Collecte EuroVision 2016-2022	24
Chapitre 3 : Préparation des données	25
I. Sélection des variables	25
1. Contestants	25
2. Votes	25
3. Collecte EuroVision 2016-2022	26
II. Prétraitement des données	26
1. Contestants	26
2. Votes	26
3. Collecte EuroVision 2016-2022	26

III. Construction des données.....	27
1. Analyse de sentiment.....	27
2. Feature engineering	30
Chapitre 4 : Modélisation	32
I. Choix de modèle.....	32
II. Optimisation des modèles	32
Chapitre 5 : Évaluation	35
I. Évaluation des résultats.....	35
II. Modèle approuvé.....	37
Chapitre 6 : Déploiement	38
I. Processus de déploiement	38
II. Développement de l'application web	39
III. Présentation de l'application WEB.....	40
1. Page d'accueil.....	40
2. Page de prédiction	42
Conclusion et perspectives	44
Annexe 1	45
Annexe 2	49

Liste des Figures

Figure 1 : Extrait de l'ensemble de données "Contestants"	13
Figure 2 : Extrait de l'ensemble de données "Votes"	13
Figure 3 : Schéma du processus de scraping	15
Figure 4 : Extrait de la collecte "EuroVision 2016-2022"	17
Figure 5 : Fréquence d'être qualifié pour la finale entre 1956 et 2022	20
Figure 6 : Moyenne des points_final_total par pays entre 1956 et 2022.....	20
Figure 7 : Moyenne des points collectés lors de la grande finale par pays entre 1956 et 2022	21
Figure 8 : Nuage de mots des tweets collectés	22
Figure 9 : Tableau des valeurs manquantes (contestants)	23
Figure 10 : Tableau des valeurs manquantes (Votes).....	23
Figure 11 : Exemples de textes bruts (Tweets).....	24
Figure 12 : Processus de nettoyage et feature engineering.....	26
Figure 13 : Schéma du processus de Transfer Learning effectué.....	28
Figure 14 : Comparaison du stemming et de la lemmatization	29
Figure 15 : Comparaison des performances des différents modèles entraînés.....	30
Figure 16 : Performances du modèle retenu : la Régression Logistique	30
Figure 17 : Résultat après les transformations appliquées	31
Figure 18 : Optimisation de la régression logistique.....	33
Figure 19 : Formule de la précision.....	35
Figure 20 : Formule du rappel	35
Figure 21 : Formule du F1-Score	35
Figure 22 : Performances du Random Forest	36
Figure 23 : Performances de la Régression Logistique	36
Figure 24 : Performances du OneClass-SVM	36
Figure 25 : Prédiction des probabilités avec la régression logistique.....	37
Figure 26 : Modèle de déploiement.....	38
Figure 27 : Fonctionnement de l'application Web	40
Figure 28 : Page d'accueil de l'application web	40
Figure 29 : Participants finalistes par édition	41
Figure 30 : Top 5 des pays ayant le plus gagné d'édition.....	41
Figure 31 : Evolution des votes des gagnants au fil du temps.....	42
Figure 32 : Prédiction de l'opinion publique sur Twitter des participants par édition	42
Figure 33 : Prédiction des probabilités de victoire par édition et par pays	43

Liste des abréviations

UTT : Université de Technologie de Troyes

ML : Machine Learning

RL : Régression Logistique

RF : Random Forest

SVM : Support Vector Machine

SNS : Social Network Service

API : Application Programming Interface

CV : Cross Validation

NLP : Natural Language Processing

Introduction générale

Twitter compte plus de 340 millions d'utilisateurs actifs, et leur participation a conduit à la génération rapide de données, en particulier dans le contexte de sujets populaires tels que les reportages, la politique et le sport. Ce réseau est devenu l'un des médias les plus populaires sur les réseaux sociaux aujourd'hui. Twitter est donc logiquement devenu un outil de recherche riche pour résoudre divers problèmes de sciences sociales et de science des données.

Il a été utilisé avec succès comme source de données pour l'analyse de texte, l'exploration de sentiments et d'opinions, la modélisation de sujets, la classification et la synthèse de texte, etc.

Les utilisateurs de Twitter partagent des informations opportunes sur divers événements en cours, reflétant généralement leurs opinions personnelles, leurs réactions émotionnelles et leurs points de vue controversés. Presque toute personne impliquée ou suivant l'événement peut partager des informations en temps réel. Par conséquent, au fur et à mesure que l'événement se déroule, ces informations peuvent atteindre n'importe où dans le monde. Ainsi, les médias sociaux peuvent être considérés comme une source précieuse d'informations à jour générées par les groupes d'utilisateurs dans le contexte de presque tous les événements.

L'objectif de ce projet est de développer et d'évaluer un système qui prédit le gagnant de l'émission EuroVision en se basant sur l'approche CRISP-DM.

Tout d'abord, j'utilise plusieurs techniques pour nettoyer les textes collectés à partir du réseau TWITTER. Ensuite, une classification des sentiments est utilisée pour diviser le flux de données d'entrée en deux flux indépendants (positif et négatif). Puis, je propose trois modèles de classification afin de prédire le vainqueur de l'édition 2022.

Finalement, je résume et présente des perspectives de chacune de mes réalisations.

Chapitre 1 : Compréhension du métier

Introduction

La première étape de chaque projet basé sur l'approche CRISP-DM est la compréhension du métier, qui se concentre sur la compréhension des objectifs et des exigences du projet. En résumé, ce seront les piliers des prochaines étapes de ce projet. Je terminerai le premier chapitre avec les différents outils et technologies utilisés.

I. Objectif métier

Comment prédire le gagnant de l'Eurovision à partir des réactions des auditeurs de l'émission sur le réseau Twitter ?

II. Evaluation de la situation

Cette étape m'aide à analyser la situation actuelle du projet en identifiant les ressources et les parties prenantes du projet.

Dans mon cas, je dois découvrir les caractéristiques qui favorisent le gain d'une compétition et vérifier s'il existe des caractéristiques communes aux réactions des téléspectateurs.

III. Objectif data mining

Les objectifs techniques sont parallèles aux objectifs métiers. Rien ne peut se faire sans l'autre. J'ai donc utilisé différentes techniques :

- Collecte des données : Utiliser le réseau social Twitter afin de récolter un maximum de données.
- Prétraitement et extraction des caractéristiques : Cette étape consiste à nettoyer les données fournies afin de leur donner du sens et de les rendre plus pertinentes.
- Machine Learning : Cela se traduit par l'application de plusieurs modèles d'apprentissage pour la classification des données et la prédiction.
- Visualisation : Des visualisations qui résument les données et aident à mieux comprendre.

IV. Plan du projet

Cette partie vise à décrire les principales étapes qui auraient lieu au cours de ce projet. De plus, je parlerai des différents outils et technologies utilisés pour atteindre les objectifs d'exploration de données et l'objectif métier.

1. Étapes de planification du projet

Comme précisé ci-dessus, le plan qui sera utilisé dans ce projet est CRISP-DM qui, grâce à sa spécification agile, peut me faire atteindre mes objectifs facilement.

Business Understanding

- ◆ Définition du contexte
- ◆ Formulation de la problématique
- ◆ Présentation du sujet

DATA Understanding

- ◆ Description de l'entité tweets
- ◆ Réaliser un tableau de comparaison des APIs Twitter
- ◆ Choisir la manière de stocker les données (tweets) récoltés
- ◆ Déterminer une liste de règles permettant de filtrer les tweets
- ◆ Récolter les données des sessions passées de l'Eurovision
- ◆ Pipeline depuis Python vers la solution choisie pour stocker les données
- ◆ Description des données
- ◆ Exploration des données
- ◆ Vérification de la qualité des données

Data Preparation

- ◆ Choisir les champs utiles à récupérer issue des tweets renvoyés par l'API
- ◆ Sélectionner les colonnes pertinentes
- ◆ Nettoyer les données

- ◆ Traduire les données
- ◆ Features engineering et sentiment analysis
- ◆ Création de jeux de données (Train/Test)

Modeling

- ◆ Choix des modèles ML
- ◆ Entraînement des modèles
- ◆ Optimisation des paramètres (GridSearch, CV)


Evaluation

- ◆ Test du modèle
- ◆ Evaluation du modèle

Deployment

- ◆ Choix d'un support de visualisation
- ◆ Visualisation sur les données sources et les résultats

2. Outils et technologies utilisés

Outils, technologies et langage	Définitions
 <p>Visual Studio Code</p>	<p>Éditeur de code source autonome qui s'exécute sur Windows, macOS et Linux. Le meilleur choix pour JavaScript et les développeurs web, avec des extensions pour prendre en charge à peu près n'importe quel langage de programmation. Dans ce cas, j'ai utilisé Python.</p>

<p>Python (Environnement : Jupyter kernel)</p> 	<p>Un langage de programmation objet, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.</p>
 <p>Streamlit</p> <p>Streamlit</p>	<p>C'est une librairie open source Python créée en 2018. Elle offre une alternative très intéressante pour la construction et le partage des <i>web apps</i> et permet de créer du <i>front-end</i> de manière innovante : la partie <i>front-end</i> est généralement développée en langages HTML, CSS et JavaScript tandis que Streamlit ne requiert que la connaissance du langage Python.</p>
<p>Microsoft Teams</p> 	<p>C'est un hub de collaboration qui reprend les fonctionnalités de Microsoft suivantes :</p> <p>Exchange : logiciel de travail de groupe associé à une messagerie électronique</p> <p>SharePoint : application permettant le partage d'informations entre plusieurs utilisateurs.</p>


<p>Github</p> 	<p>C'est un service de gestion d'hôte virtuel et de développement logiciel qui permet à ses utilisateurs de gérer et stocker leurs projets. Il assure également le suivi et le contrôle des modifications qui y sont apportées.</p>
---	---

Tableau 1 : Outils et technologies utilisés

Conclusion

L'étape de compréhension du métier et l'ensemble du domaine m'aide à m'immerger dans cette étude. Cela me permet d'aborder les prochaines étapes solides et bien informées dans le domaine. Sans ce passage, tout ce qui va suivre sera plus difficile à attaquer.

Chapitre 2 : Compréhension des données

Introduction

Lors de l'élaboration de la méthodologie CRISP, une question s'est posée : quelles données j'aurai besoin ? C'est propre ? Et maintenant il faut répondre. Pour aller plus loin, l'étape de collecte de données sera nécessaire. Je finirai par une description de tout ce qui m'est fourni dans ces ensembles de données afin de vérifier leurs qualités.

I. Collecte de données

Cette partie est réalisée via l'API Twitter V2. Ce dernier est un service qui permet d'envoyer des requêtes automatiques au sein des tweets et qui peut être utilisé pour surveiller ou analyser le contenu des tweets. De plus, une partie de scrapping est ajoutée pour enrichir ces données.

Dans cette étude j'ai exploité 3 ensembles de données :

1. Contestants

Le premier jeu de données est intitulé « Contestants ». Un ensemble de données qui contient le classement de la compétition pour les finales et les demi-finales entre 1956 et 2020 qui est accessible pour le grand public sur la plateforme Github sous forme d'un fichier « csv ».

Un scrapping du site officiel de l'EuroVision est fait pour compléter la collecte en rajoutant les années 2021 et 2022.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	year	to_country	to_country	performer	song	place_contest	sf_num	running_final	running_sf	place_final	points_final	place_sf	points_sf	poi
1481	2018	cy	Cyprus	Eleni Foureira	Fuego	2	1.0	25.0	19.0	2.0	436.0	2.0	262.0	253
1482	2018	at	Austria	Cesár Samp	Nobody But	3	1.0	5.0	13.0	3.0	342.0	4.0	231.0	71.0
1483	2018	de	Germany	Michael Schenker	You Let Me V	4		11.0		4.0	340.0			136
1484	2018	it	Italy	Ermal Meta	Non Mi Avet	5		26.0		5.0	308.0			249
1485	2018	cz	Czech Repu	Mikolas Jose	Lie To Me	6	1.0	14.0	5.0	6.0	281.0	3.0	232.0	215
1486	2018	se	Sweden	Benjamin Ing	Dance You O	7	2.0	20.0	15.0	7.0	274.0	2.0	254.0	21.0
1487	2018	ee	Estonia	Elina Nechay	La Forza	8	1.0	6.0	9.0	8.0	245.0	5.0	201.0	102
1488	2018	dk	Denmark	Rasmussen	Higher Grou	9	2.0	15.0	5.0	9.0	226.0	5.0	204.0	188
1489	2018	md	Moldova	DoReDoS	My Lucky Da	10	2.0	19.0	7.0	10.0	209.0	3.0	235.0	115
1490	2018	al	Albania	Eugenit Bush	Mall	11	1.0	12.0	3.0	11.0	184.0	8.0	162.0	58.0
1491	2018	lt	Lithuania	Ieva Zasima	When We're	12	1.0	4.0	6.0	12.0	181.0	9.0	119.0	91.0
1492	2018	fr	France	Madame Mo	Mercy	13		13.0		13.0	173.0			59.0
1493	2018	bg	Bulgaria	Equinox	Bones	14	1.0	18.0	10.0	14.0	166.0	7.0	177.0	66.0
1494	2018	no	Norway	Alexander R	That's How Y	15	2.0	7.0	1.0	15.0	144.0	1.0	266.0	84.0
1495	2018	ie	Ireland	Ryan O'Shau	Together	16	1.0	24.0	18.0	16.0	136.0	6.0	179.0	62.0
1496	2018	ua	Ukraine	M&A lovin	Under the La	17	2.0	1.0	18.0	17.0	130.0	6.0	179.0	119
1497	2018	nl	Netherlands	Waylon	Outlaw In 'Er	18	2.0	23.0	8.0	18.0	121.0	7.0	174.0	32.0
1498	2018	rs	Serbia	Sanja Ilia	Novo Deca	19	2.0	10.0	3.0	19.0	113.0	9.0	117.0	75.0
1499	2018	au	Australia	Jessica Maut	We Got Love	20	2.0	16.0	9.0	20.0	99.0	4.0	212.0	9.0

Figure 1 : Extrait de l'ensemble de données "Contestants"

2. Votes

Le deuxième jeu de données est intitulé « Votes ». Un ensemble de données qui contient les données de vote de pays à pays entre 1957 et 2019 qui est accessible pour le grand public sur la plateforme Github sous forme d'un fichier « csv ».

De même, le scrapping du site officiel de l'Eurovision est réalisé pour compléter la collecte en rajoutant les années 2021 et 2022.

	A	B	C	D	E	F	G	H	I	J
1	year	round	from_country	to_country	from_country	to_country	total_points	tele_points	jury_points	
48014	2021	final	gb	no	gb	no	2	0	0	
48015	2021	final	gb	be	gb	be	1	0	1	
48016	2021	final	gb	az	gb	az	0	0	0	
48017	2021	final	gb	al	gb	al	0	0	0	
48018	2021	final	gb	sm	gb	sm	3	0	3	
48019	2021	final	gb	nl	gb	nl	0	0	0	
48020	2021	final	gb	es	gb	es	2	0	2	
48021	2021	final	gb	de	gb	de	0	0	0	
48022	2021	final	gb	gb	gb	gb	0	0	0	
48023	2021	semi-final-1	au	mt	au	mt	22	10	12	
48024	2021	semi-final-1	au	ua	au	ua	16	12	4	
48025	2021	semi-final-1	au	ru	au	ru	14	7	7	
48026	2021	semi-final-1	au	lt	au	lt	10	8	2	
48027	2021	semi-final-1	au	il	au	il	12	4	8	
48028	2021	semi-final-1	au	cy	au	cy	16	6	10	
48029	2021	semi-final-1	au	se	au	se	6	0	6	
48030	2021	semi-final-1	au	az	au	az	6	1	5	
48031	2021	semi-final-1	au	be	au	be	0	0	0	
48032	2021	semi-final-1	au	no	au	no	4	3	1	
48033	2021	semi-final-1	au	hr	au	hr	5	5	0	
48034	2021	semi-final-1	au	no	au	no	0	0	0	

Figure 2 : Extrait de l'ensemble de données "Votes"

3. Scraping des années manquantes

Cependant, ces deux datasets ne contiennent pas toutes les données nécessaires à mon analyse. En effet, ils ne contiennent que les données allant jusqu'à l'édition 2020 pour les "contestants" et 2019 pour les "votes", étant donné l'annulation de l'édition 2020 en raison de la pandémie. Pour pallier ces données manquantes, j'ai décidé de réutiliser le code présent sur le Github des jeux de données pour l'adapter à mes besoins et en corrigeant les potentielles erreurs.

Pour réaliser ce scraping j'ai alors utilisé les bibliothèques **BeautifulSoup** et **Selenium**. Le processus suivi est le suivant :

- Depuis la page sur l'eurovision je me rends à la page correspondant à première année passée en paramètre du script (dans mon cas 2016)
- Depuis cette page on parcourt le tableau des résultats de l'année
- On y récupère alors les informations suivantes :
 - Année
 - Ordre de passage
 - Pays
 - Musique
 - Interprète
 - Points reçus
 - Place en finale
- Une fois ces données récupérées on parcourt le tableau des votes, on y récupère :
 - Année
 - Pays votant
 - Pays recevant
 - Nombre de points donné
- Après cela, on retourne à la page précédente avec les résultats des différentes années et l'on répète le processus pour l'année suivante (dans mon cas 2022)

Dès lors que toutes les informations ont été collectées, on sauvegarde le tout dans des fichiers .csv

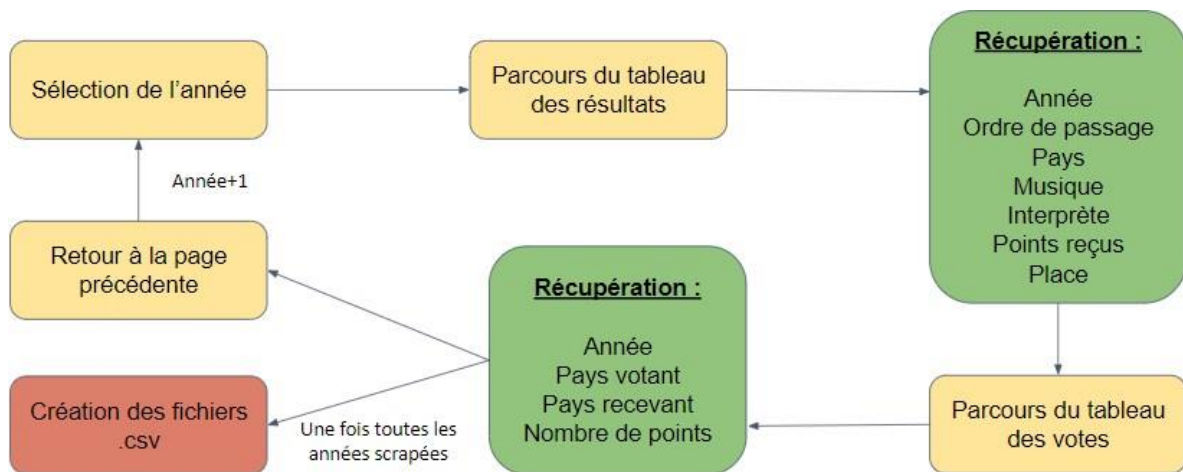


Figure 3 : Schéma du processus de scraping

4. Collecte EuroVision 2016-2022

Dans le but d'essayer de prédire le gagnant d'une certaine édition de l'Eurovision en se basant sur les avis des téléspectateurs sur Twitter, il est nécessaire de collecter et d'analyser ces avis. Pour ce faire, j'ai commencé en comparant les différents moyens d'accéder aux données Twitter et plus particulièrement aux messages postés par les téléspectateurs.

i. Technologie utilisée

La solution la plus évidente consistait à utiliser l'API de ce même réseau social. J'ai donc comparé les différentes versions de cette API (V1, V2) et j'ai étudié les différentes librairies Python permettant d'utiliser cette API. J'ai effectué plusieurs tests peu concluants avec la librairie Tweepy. L'échec de ces différents essais résultait en réalité de l'utilisation de l'API en elle-même et non du choix d'une bibliothèque Python en particulier. En effet, l'API Twitter imposait un nombre de restrictions importantes qui ne me permettait pas de collecter les tweets désirés correctement.

Premièrement, l'API imposait un nombre de requêtes maximum sur un laps de temps assez réduit. De plus, le nombre de tweets récoltés par requête exécutée était également limité. Ces deux restrictions ne me permettaient

pas de récolter une quantité de données suffisante. La taille en nombre de caractère de chaque requête était également limitée : je ne pouvais pas construire des requêtes suffisamment détaillées et les tweets récoltés n'étaient pas assez précis à mon goût. Enfin, L'API Twitter ne m'autorisait à récolter des tweets ne datant que des 7 derniers jours. Cette restriction m'a conforté dans le choix d'abandonner l'utilisation de cette solution.

Je me suis donc tournée vers une solution plus accessible : SNScraper. Social Network Scraper est un scraper automatique black box permettant de récolter des données depuis le réseau social Twitter, Instagram, Facebook et bien d'autres avec très peu de limitations. La librairie Python associée me permet de collecter des tweets sans limitation du nombre de requêtes ou encore de la temporalité des données ciblées. En revanche, la collecte est un peu plus chronophage en utilisant cette méthode qu'en utilisant l'API.

Le système de collecte de la librairie est basé sur la sémantique de requête avancée du réseau Twitter, en agrégeant des hashtags, des mots clés et des ponctuations précises afin de donner un sens à une suite de mots. Ainsi, il est possible de récolter des tweets contenant obligatoirement ou non des hashtags, des mots clés, des groupes de mots précis, ou en les excluant entre deux dates bien précises.

	API Twitter	SNScraper
Nombre de requêtes	Limité	Illimité
Tweets par requêtes	Limité	Illimité
Longueur des requêtes	Limitée	Illimitée
Temporalité des données	7 derniers jours	Au choix
Temps de collecte	Moyen	Long

Tableau 2 : Comparatif des solutions de collecte de Tweets

Pipeline de collecte

Je vais maintenant vous présenter le pipeline de collecte utilisée afin de récolter les avis des internautes sur Twitter.

```
{
  "year": "2019",
  "country": "Spain",
  "points_final": 54.0,
  "place_final": 22.0,
  "winner": true,
  "date": "2019-05-18 22:37:33+00:00",
  "content": "Miki Espa\u00f1a entera sabe que eres ganador #Eurovision #Eurovision2019 #EurovisionRTVE",
  "hashtags": [
    "Eurovision",
    "Eurovision2019",
    "EurovisionRTVE"
  ],
  "retweet_count": 0,
  "like_count": 0,
  "user_username": "FighterShowho",
  "user_verified": false,
  "user_followers_count": 371
},
{
```

Figure 4 : Extrait de la collecte "EuroVision 2016-2022"

II. Description des données

1. Contestants

Cet ensemble se compose de 1 683 lignes et 21 colonnes où chaque ligne contient des informations relatives aux différents participants entre 1956 et 2022.

Pour chaque candidat il y a :

year: année du concours

to_country_id: identifiant de pays du candidat

to_country: nom du pays du candidat

performer: nom du candidat

song: titre de la chanson du candidat

sf_num: a participé à la demi-finale 1, 2 ou 0 (de 2004 à 2008, il n'y a eu qu'une seule demi-finale)

running_final: dans la diffusion de la finale du concours

running_sf: ordre dans la diffusion de la demi-finale du concours

place_final: place dans la finale

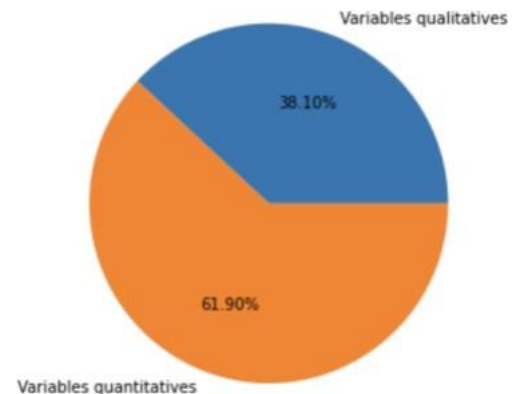
points_final: points dans la finale

place_sf: place en demi-finale

points_sf: points en demi-finale

points_tele_finale: points de télévote dans la finale du concours
 points_jury_final: points de vote du jury dans la finale du concours
 points_tele_sf: points de télévote en demi-finale du concours
 points_jury_sf: points de vote du jury en demi-finale du concours
 composers: Compositeur
 lyricists: Auteur des paroles
 lyrics: paroles de la chanson
 youtube_url: url vers la vidéo sur YouTube

Dans ce jeu de données, il y a 8 variables catégorielles soit 38.1 % et 13 variables quantitatives soit 61.9 %.



2. Votes

Ce fichier se compose de 50 365 lignes et 9 colonnes où chaque ligne présente le vote d'un pays à un autre lors de la grande finale entre 1956 et 2022.

Pour chaque édition il y a:

year : année du concours

round : finale, demi-finale

from_country_id: identifiant de pays du pays donnant des points

to_country_id: identifiant de pays du pays recevant des points

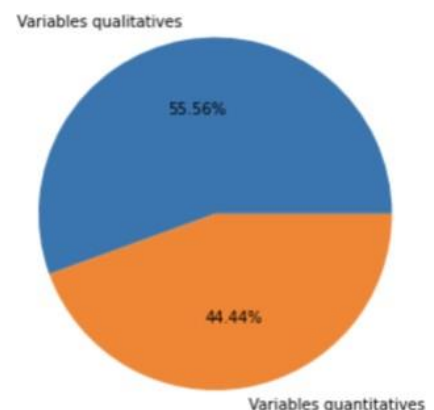
from_country : nom du pays donnant des points

to_country: nom du pays recevant des points

total_points : nombre de points attribués

tele_points: points de télévote dans la finale du concours

jury_points: points de vote du jury dans la finale du concours



Dans ce jeu de données, il y a 5 variables catégorielles soit 55.56 % et 4 variables quantitatives soit 44.44 %.

3. Collecte EuroVision 2016-2022

Cette collecte se compose de 77 501 lignes et 13 colonnes où chaque ligne est un tweet contenant les réactions et les informations de l'utilisateur de Twitter.

Pour chaque tweet il y a :

year: année du concours

country: nom du pays du candidat

points_final: :points collectés dans la finale

place_final: place dans la finale

winner: si le candidat est vainqueur (true /false)

date: la date du tweet

content : le texte du tweet

hashtags : tous les hashtags qui existent dans le tweet

retweet_count: le nombre de retweet

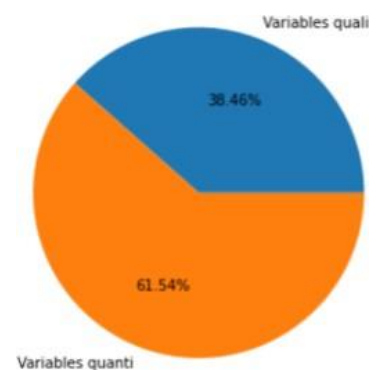
like_count: le nombre de j'aime

user_username: le nom de l'utilisateur

user_verified: utilisateur vérifié

user_followers_count: le nombre des abonnés de l'utilisateur

Dans ce jeu de données, il y a 8 variables catégorielles soit 61.56 % et 5 variables quantitatives soit 38.46 %.



III. Exploration des données

1. Contestants

Dans ce concours, on remarque que l'Allemagne, la France, le Royaume Uni, l'Espagne et le Suède sont les Top 5 pays les plus souvent qualifiés pour la finale EuroVision entre 1956 et 2022. (Voir figure 4)

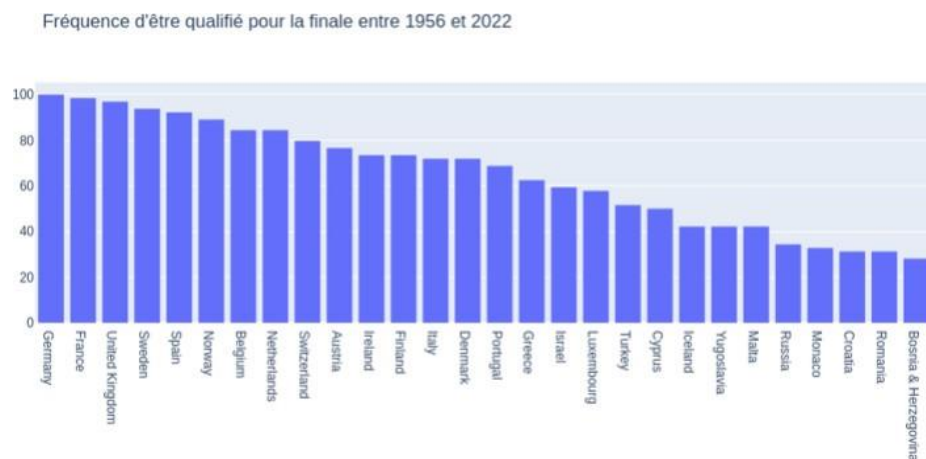


Figure 5 : Fréquence d'être qualifié pour la finale entre 1956 et 2022

La Russie a la moyenne la plus importante des points_final_total collectés, en deuxième place le Portugal et puis la Suisse. (Voir figure 5)

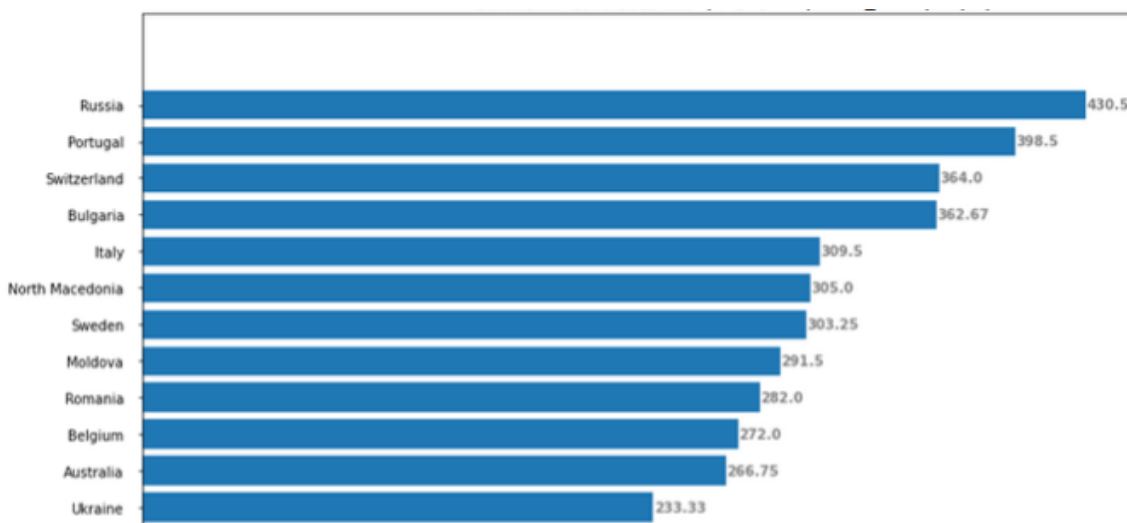


Figure 6 : Moyenne des points_final_total par pays entre 1956 et 2022

2. Votes

Ici, on remarque que la Russie a la moyenne la plus élevée des points collectés lors de la grande finale suivie par Ukraine et Italie. (Voir figure 6)

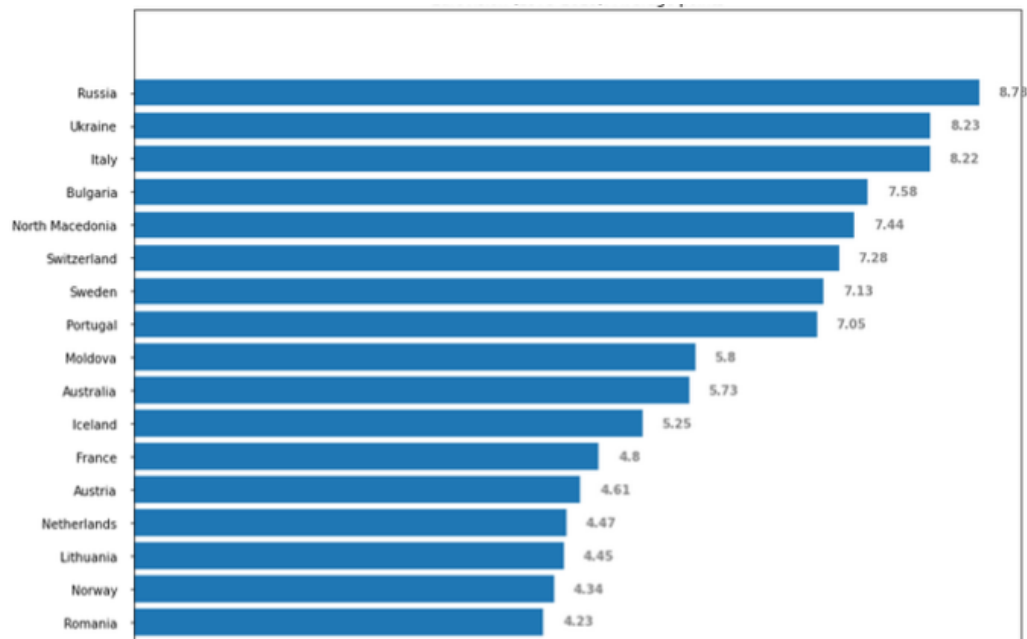


Figure 7 : Moyenne des points collectés lors de la grande finale par pays entre 1956 et 2022

3. Collecte EuroVision 2016-2022

Ce nuage de mots résultant m'a permis d'arriver à quelques conclusions :

- Le mot « Eurovision » est le plus dominant, ce qui est logique, car c'est le nom de l'émission télévisée.
- Plusieurs noms de pays : « Ukraine, Italie, Sweden, France, Norway, Germany. », ce qui est évident puisque ce sont les pays participants.
- plusieurs mots à intonation positive, puissante et encourageante : « win, go, like, well perform, top, good, »



Figure 8 : Nuage de mots des tweets collectés

IV. Vérification de la qualité des données

1. Contestants

Hors les colonnes (year; to_country_id ;to_country ;performer; song et lyrics), il y a plusieurs données manquantes. En particulier, pour les variables place_sf ; points_sf; points_tele_final ; points_jury_final ; points_tele_sf ; points_jury_sf parce que ces dernières commencent à partir de l'année 2016. (Voir figure 9)

	Missing Values	% of Total Values
points_jury_sf	1538.0	91.4
points_tele_final	1527.0	90.8
points_jury_final	1510.0	89.8
points_tele_sf	1470.0	87.4
place_sf	1160.0	69.0
running_sf	1109.0	65.9
points_sf	1109.0	65.9
sf_num	1074.0	63.9
lyricists	673.0	40.0
place_final	362.0	21.5
points_final	306.0	18.2
running_final	293.0	17.4
place_contest	52.0	3.1
composers	42.0	2.5

Figure 9 : Tableau des valeurs manquantes (contestants)

2. Votes

Pas de votes pour l'année 2020 à cause de la pandémie.

Pour les colonnes tele_points;jury_points, il y 82,5% de données manquantes et les valeurs commencent à partir de l'année 2016.

Pour le reste des colonnes, il n'y a pas grande chose à dire, c'est très propre. (Voir figure 10)

	Missing Values	% of Total Values
tele_points	30136.0	82.5
jury_points	30136.0	82.5

Figure 10 : Tableau des valeurs manquantes (Votes)

3. Collecte EuroVision 2016-2022

Dans cette collecte, j'ai des textes bruts multilingues contenant beaucoup d'aléatoires influençant l'estimation des modèles : accents, lettres minuscules, signes de ponctuation, caractères spéciaux, des émojis, des adresses url etc ... (voir figure 10)

Now that #Amsterdam has pulled itself out of the city candidates to host the #Eurovision2019, which city should host the #EurovisionSongContest2020 in #TheNetherlands🇳🇱? #Rotterdam2020, #Arnhem2020, #Maastricht2020, #Utrecht2020 or #DenBosch2020? #TheNetherlands2020 #Eurovision

Straks weten we wie Cyprus zal vertegenwoordigen in Rotterdam! #Eurovision #ESC2021 #OpenUp bit.ly/2IWWlyf

Entra nel nuovo gruppo Facebook dedicato all'#Eurovision Song Contest. Discussioni, notizie, curiosità, sondaggi, domande e risposte sull'evento musicale più seguito al mondo. #EurovisionInside è qui > escne.ws/escinside

Figure 11 : Exemples de textes bruts (Tweets)

Conclusion

Il est vrai que tout au long de cette étape, j'ai réussi à comprendre mes données et à préparer tout le terrain favorable pour le travail. Mais tout ce que j'économise n'est pas suffisant pour faire une prédiction parfaitement précise, car toutes les données ne sont pas pertinentes. Il y a des textes bruts qui contiennent beaucoup d'aléatoire qui affectent l'estimation des modèles : les accents, les minuscules, les signes de ponctuation, les caractères spéciaux... Tout cela m'amène à une prochaine étape, très importante dans projets de science des données qui est la préparation de données

Chapitre 3 : Préparation des données

Introduction

Comme mentionné dans le chapitre précédent, mon problème repose sur les données de sentiment fournies par les réseaux sociaux Twitter. Ainsi, ces entrées sont en langage naturel et nécessitent un traitement spécifique à travers différentes tâches : tokenisation, normalisation de mots, lemmatisation et stemming... La partie suivante est la partie de traduction des tweets pour terminer avec l'analyse de sentiments et le feature engineering.

I. Sélection des variables

Dans cette partie, je vais citer les différentes étapes afin de sélectionner les variables que j'utiliserai dans les prochaines étapes.

1. Contestants

- Supprimer les données des années inférieures ou égales à l'année 2015 et commencer à partir de l'année 2016
- Sélectionner les colonnes qui m'intéressent : "year", "to_country_id", "to_country", "performer", "song", "place_contest", "running_final", "place_final", "points_final".
- Supprimer toutes les colonnes en rapport avec la demi-finale

2. Votes

- Supprimer les données des années inférieures ou égales à l'année 2015 et commencer à partir de l'année 2016
- Sélectionner les colonnes qui m'intéressent : "year", "round", "from_country_id", "to_country_id", "from_country", "to_country", "total_points".
- Supprimer toutes les colonnes en rapport avec la demi-finale

3. Collecte EuroVision 2016-2022

Aucun changement n'est effectué, la sélection est déjà faite lors du scrapping.

II. Prétraitement des données

Ci-dessous, les différentes étapes effectuées pour nettoyer les données afin qu'elles soient prêtes pour la modélisation.

1. Contestants

- Supprimer toutes lignes de l'année 2020 (pandémie)
- Supprimer tous les participants non qualifiés pour la finale : `running_finale > 26 and place_contest=null`

2. Votes

- Supprimer toutes lignes de l'année 2020 (pandémie)

3. Collecte EuroVision 2016-2022

Afin de répondre à mon objectif, il est nécessaire de construire un modèle d'apprentissage capable de prédire le gagnant d'une édition de l'eurovision. L'apprentissage de ce modèle nécessitera de lui proposer des données en entrée. Or les données dont je dispose actuellement ne sont pas complètes et sont de moyenne qualité.

Je décide donc de passer par 3 étapes intermédiaires :

- La traduction des tweets
- Le nettoyage des tweets
- La construction de features appropriées à la prédiction d'un gagnant par un modèle.

Cette dernière étape comprend l'analyse de sentiment des tweets afin d'en déterminer la polarité.



Figure 12 : Processus de nettoyage et feature engineering

La traduction des tweets consiste à uniformiser la langue des textes de ceux-ci afin de rendre l'analyse de sentiment plus précise et efficace. En effet, l'Eurovision étant un spectacle international, les avis récoltés se trouvaient être des textes multilingues. J'ai utilisé l'API de google translate et cette étape a été la plus chronophage de toute car la traduction de ces 77500 tweets a été effectuée en 26 heures environ.

La seconde étape consiste à nettoyer entièrement les tweets récoltés et à préparer le texte de ceux-ci pour la dernière étape. Il était donc nécessaire d'épurer et de corriger les données en ma possession. La phase de nettoyage des tweets se déroule sur plusieurs étapes comme suit:

- Écrire tout en minuscule
- Suppression des retours à la ligne
- Suppression des pseudo Twitter (exp : utilisateur @EuroVisionFrance)
- Suppression d'adresse url (modèle 'http :')
- Suppression des mentions Twitter (exp : @EuroVision)
- Suppression des hashtags (exp : #EuroVision2018)
- Suppression des émojis
- Suppression des chiffres et des caractères spéciaux (: « , ; : ! ? ' & \$ % () [] + - * / »)
- Appliquer la lemmatisation et le stemming
- Suppression des stop-words
- Tokenization et suppression des short-words

Les données sont maintenant propres. Je peux passer à la construction de features intéressantes à la prédiction d'un gagnant.

III. Construction des données

1. Analyse de sentiment

Cette étape consiste à déterminer la polarité de chacun de ces tweets à partir de son contenu maintenant nettoyé et épuré. Je veux pour cela utiliser un modèle d'apprentissage capable de prédire cette variable.

i. Transfert Learning

Or il m' est impossible d'entraîner un modèle sur mon propre jeu de données car celui-ci n'est pas labellisé. Je vais donc utiliser une méthode de transfert learning. Le transfer learning consiste en l'entraînement un modèle sur une partie d'un jeu de données adapté c'est à dire un jeu de données labellisé et reprenant des features équivalentes à mon problème.

L'évaluation des performances de ce modèle est calculée sur l'autre partie de ce même jeu de données. Si les performances sont convaincantes, on peut alors utiliser le modèle entraîné sur un jeu de données similaire mais différent du premier et ainsi prédire la variable souhaitée.

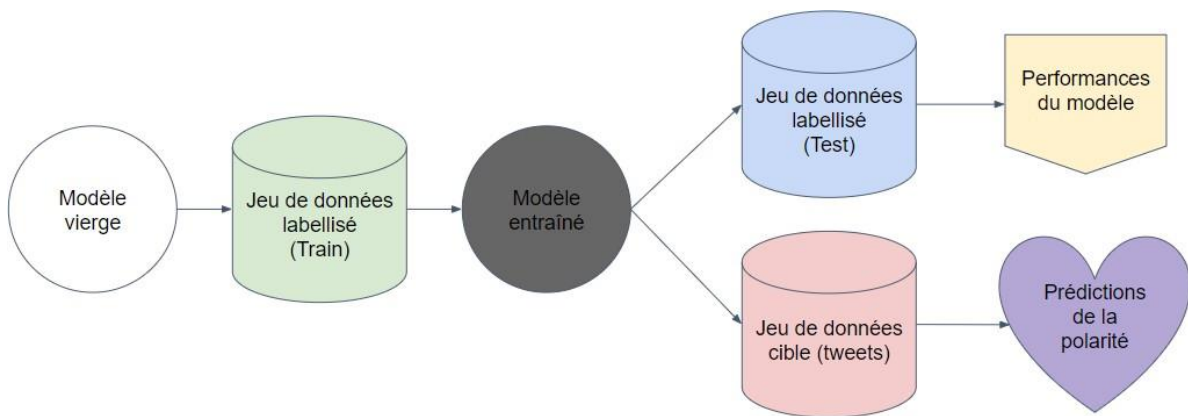


Figure 13 : Schéma du processus de Transfer Learning effectué

Pour cela, j'ai utilisé le jeu de données sentiment140 composés de 1.6 millions de tweets labellisés selon leur polarité. Je l'ai séparé en un sous jeu d'entraînement et un sous jeu de test. J'ai entraîné 3 types de modèles différents sur le jeu d'entraînement et J'ai comparé leur performance sur le jeu de test. Une fois un modèle retenu, je l'ai utilisé sur mon jeu de données cible contenant ces tweets nettoyés afin de prédire leur polarité. Dans ce jeu de données, la variable cible est donc le sentiment et se présente sous plusieurs classes :

- 0 : Sentiment négatif
- 1 : Sentiment positif

ii. Preprocessing du texte

Ce dernier contient également d'autres classes telles que l'id de l'utilisateur, la date du tweet... mais dans mon cas je ne vais utiliser que la variable cible et le texte du tweet en lui-même. Une fois cette étape de sélection, s'ensuit plusieurs étapes de NLP, Natural Language Processing, pour préparer l'entraînement du modèle qui prédira le sentiment de mes tweets.

Premièrement, j'ai retiré tous les "stopwords", c'est-à-dire les mots qui se répètent souvent et qui n'apportent pas de réel sens à un texte ainsi que toute la ponctuation. Ensuite, j'ai retiré tous les caractères qui se répètent, les URL, les nombres, ainsi que les emojis.

Pour la suite du pré-traitement des tweets j'ai utilisé "nltk" qui est une bibliothèque python pour le NLP. Celle-ci m'a permis de "tokenizer" le texte, c'est-à-dire supprimer les espaces et convertir chaque tweet en une liste de mots. Après cela, j'ai utilisé le "PorterStemmer" et le "WordNetLemmatizer" que propose cette bibliothèque pour homogénéiser les mots que l'on retrouve dans les différentes listes. En effet, le "stemming" permet de couper les mots ayant une même racine à la même position dans le mot tandis que la "lemmatization" consiste à ramener tous les mots similaires à une même racine sémantique. La figure ci-dessous résume ces deux concepts.



Figure 14 : Comparaison du stemming et de la lemmatization

iii. Entraînement du modèle pour l'analyse de sentiment

Une fois tous mes tweets pré-traités, je peux passer à la phase d'entraînement des différents modèles. Les 3 modèles que j'ai testés sont :

- Le Naive Bayes
- Le Support Vector Machine
- La régression logistique.

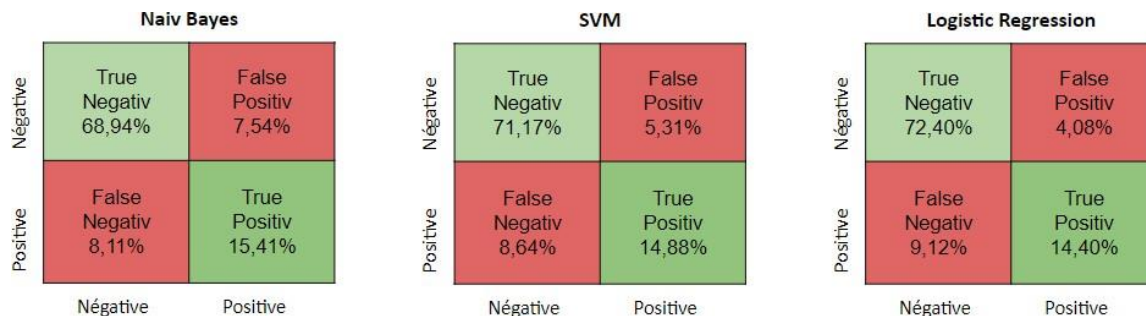


Figure 15 : Comparaison des performances des différents modèles entraînés

Le modèle retenu par ses performances et la régression logistique avec une accuracy de 0.86, un recall de 0.87 et un F1-score de 0.86. Vous pouvez visualiser ci-dessus les matrices de confusion des prédictions des 3 modèles que j'ai entraîné. On peut voir que les performances du SVM sont légèrement moins bonnes que celles de la régression. Le Naive Bayes, lui, se fait largement concurrencer. L'évaluation finale du modèle de régression logistique est la suivante :

	precision	recall	f1-score	support
0	0.89	0.95	0.92	40097
1	0.78	0.61	0.69	12332
accuracy			0.87	52429
macro avg	0.83	0.78	0.80	52429
weighted avg	0.86	0.87	0.86	52429

Figure 16 : Performances du modèle retenu : la Régression Logistique

2. Feature engineering

Le feature engineering fait référence au processus d'utilisation des connaissances du domaine pour sélectionner et transformer les variables les plus pertinentes à partir de données brutes. Une fois le nettoyage, la traduction et l'analyse faite, j'ai

éliminé les variables les moins pertinentes entre autres la date puisque j'ai conservé l'année de l'édition et les hashtags et le texte traduit car ayant une analyse de sentiment dessus.

Afin d'obtenir les variables les plus pertinentes pour mon modèle, j'ai appliqué une série d'agrégation sur mes données telles que le calcul de la somme, la médiane, le ratio ...

La figure ci-dessous montre le résultat obtenu après les transformations.

year	country	retweet_cou	like_count	user_verified	user_followers_count	positive_tweets_percentage	negative_tweets_percentage	winner
2016	Ukraine	283	803	9	288.0	87.4	12.6	1
2016	Australia	167	589	15	472.5	77.6	22.4	0
2016	Russia	107	578	2	235.0	85.8	14.2	0
2016	Bulgaria	74	303	5	293.5	84.4	15.6	0
2016	Sweden	72	367	6	254.0	81.6	18.4	0
2016	France	110	427	4	305.5	83.0	17.0	0
2016	Armenia	94	361	7	293.0	82.6	17.4	0
2016	Poland	106	374	6	230.5	81.4	18.6	0
2016	Lithuania	181	570	7	232.5	80.6	19.4	0
2016	Belgium	82	383	5	263.0	80.8	19.2	0
2016	Netherlands	98	424	7	216.0	82.6	17.4	0
2016	Malta	105	385	7	296.5	83.4	16.6	0
2016	Austria	94	376	7	292.5	82.4	17.6	0
2016	Israel	92	309	7	279.0	81.4	18.6	0
2016	Latvia	89	370	5	236.0	81.6	18.4	0
2016	Italy	90	370	5	252.5	83.0	17.0	0
2016	Azerbaijan	92	377	4	292.5	81.2	18.8	0

Figure 17 : Résultat après les transformations appliquées

Conclusion

Les outils de préparation des données m'ont permis de nettoyer les données avant de les analyser. Cela fournit une base solide et fiable pour m'aider dans les prochaines phases de mon projet, en particulier la modélisation en permettant au projet de s'exécuter de manière plus rapide et plus transparente.

Chapitre 4 : Modélisation

Introduction

La phase de modélisation fait suite à la phase de préparation. Elle permet de créer un ou plusieurs modèles pouvant m'aider à déterminer le gagnant de l'eurovision. Mon processus de création de modèle a été itératif. Il a consisté à :

- Choisir des modèles de bases
- Optimiser ces modèles

I. Choix de modèle

Je suis dans le cas d'un apprentissage supervisé car j'ai les valeurs de la variable cible à prédire : « winner » est ma colonne cible. C'est une variable qualitative binaire (1 == gagnant et 0 == perdant). Plusieurs algorithmes permettent de faire de la classification binaire. An nombre de ces algorithmes, j'ai choisi de tester ceux qui suivent :

- La régression logistique ;
- Les forêts aléatoires ;
- Le One class SVM.

De plus, il est important de souligner que je suis en présence d'une classification déséquilibrée. En effet j'ai 149 observations pour la classe 0 contre 6 pour la classe 1, ce qui représente un ratio d'environ 1/25. Afin de tenir compte de ce déséquilibre durant ma phase d'évaluation, j'utiliserai la métrique « Recall » qui me permettra de prédire au maximum toutes les observations de la classe 1.

II. Optimisation des modèles

Dans un premier temps, j'ai séparé le jeu de données en deux : il s'agit des données d'entraînement et de test. Mes données d'entraînement sont constituées des données de 2016 à 2021 et celui du test contient les données de 2022. Dans un second temps, j'ai :

- optimisé les hyper-paramètres des modèles en effectuant une recherche par grille sur un espace d'hyper-paramètres non exhaustif défini par ces soins : la sélection du meilleur jeu d'hyper-paramètres pour chacun des modèles est basé sur une validation croisée à 3 plis.
- accompagné le choix de ces hyper-paramètres en entraînant sur la métrique de rappel : en effet, je souhaite être en mesure de détecter le plus possible de gagnant, quitte à avoir un nombre de faux positifs élevé.

```
def logist_modelisation(x_train, y_train):
    # grille de valeurs
    weights = np.linspace(0.1,0.9,100)

    params = [{"C": [0.01, 0.2, 0.5, 1, 5, 10, 20],
               "penalty": ["l1", "l2"],
               'max_iter': [100, 500, 700, 900],
               "class_weight": [{0:x, 1:1.0-x} for x in weights]}]

    logitCV = GridSearchCV(
        LogisticRegression(),
        params,
        scoring="recall",
        cv=3,
        n_jobs=-1,
        return_train_score=True)

    logitCV = logitCV.fit(x_train, y_train)

    return logitCV.best_estimator_
```

Figure 18 : Optimisation de la régression logistique

J'ai les modèles définitifs suivants :

- Random Forest:

RandomForestClassifier(class_weight={0:0.407070707070707, 1:0.592929292929293}, max_features=4, n_estimators=10)

- Régression logistique :

LogisticRegression(C=0.01, class_weight={0: 0.1, 1: 0.9})

- One Class SVM:

OneClassSVM(max_iter=100, nu=0.1)

Conclusion

Cette phase m'a permis d'aboutir à 3 modèles pouvant potentiellement m'aider à résoudre ma problématique. Ces modèles seront évalués à l'étape suivante pour en garder qu'un seul pour le déploiement.

Chapitre 5 : Évaluation

Introduction

La précédente étape qui est la modélisation m'a permis d'avoir 03 modèles. Durant la phase d'évaluation, chaque modèle sera évalué puis comparé entre eux. Pour cela j'utiliserai des métriques d'évaluation adaptées à ma problématique.

I. Évaluation des résultats

Les métriques d'évaluation utilisées sont :

- Précision : parmi toutes les observations classifiées, combien ont été bien classifiées ?

$$\text{Précision} = \frac{TP}{TP + FP}$$

Figure 19 : Formule de la précision

- Rappel : parmi toutes les observations ayant gagné l'eurovision, combien sont effectivement détectées ?

$$\text{Rappel} = \frac{TP}{TP + FN}$$

Figure 20 : Formule du rappel

- F1-Score : évalue la capacité du modèle de classification à prédire efficacement les individus positifs (gagnant), en faisant un compromis entre la précision et le rappel.

$$F1-Score = \frac{TP}{TP + \frac{1}{2}(FN + FP)} = \frac{2}{\frac{1}{Précision} + \frac{1}{Rappel}}$$

Figure 21 : Formule du F1-Score

Mais un accent particulier a été mis sur la métrique de **RAPPEL** car mon objectif est de prédire le plus possible la classe 1 de ces jeux de données.

Ainsi, l'évaluation de ces modèles se présente comme suit :

Random forest :

-----Training data-----					
	precision	recall	f1-score	support	
0	0.98	1.00	0.99	125	
1	1.00	0.60	0.75	5	
accuracy			0.98	130	
macro avg	0.99	0.80	0.87	130	
weighted avg	0.98	0.98	0.98	130	
-----Test data-----					
	precision	recall	f1-score	support	
0	0.96	1.00	0.98	24	
1	0.00	0.00	0.00	1	
accuracy			0.96	25	
macro avg	0.48	0.50	0.49	25	
weighted avg	0.92	0.96	0.94	25	

Figure 22 : Performances du Random Forest

Régression Logistique :

-----Training data-----					
	precision	recall	f1-score	support	
0	0.98	0.97	0.97	125	
1	0.33	0.40	0.36	5	
accuracy			0.95	130	
macro avg	0.65	0.68	0.67	130	
weighted avg	0.95	0.95	0.95	130	
-----Test data-----					
	precision	recall	f1-score	support	
0	1.00	0.62	0.77	24	
1	0.10	1.00	0.18	1	
accuracy			0.64	25	
macro avg	0.55	0.81	0.48	25	
weighted avg	0.96	0.64	0.75	25	

Figure 23 : Performances de la Régression Logistique

One Class SVM:

-----Training data-----					
	precision	recall	f1-score	support	
0	0.00	0.00	0.00	125	
1	0.04	1.00	0.08	5	
micro avg	0.04	0.04	0.04	130	
macro avg	0.02	0.50	0.04	130	
weighted avg	0.00	0.04	0.00	130	
-----Test data-----					
	precision	recall	f1-score	support	
0	0.00	0.00	0.00	24	
1	0.04	1.00	0.08	1	
accuracy			0.04	25	
macro avg	0.02	0.50	0.04	25	
weighted avg	0.00	0.04	0.00	25	

Figure 24 : Performances du OneClass-SVM

II. Modèle approuvé

Au vu des évaluations précédemment réalisées, il en ressort que le modèle de régression logistique présente une meilleure précision sur le dataset de test qui est de 64% avec un “Recall” de 100%. De plus, ce modèle positionne le gagnant (Ukraine) dans le top 5 des probables gagnant comme l’indique la figure ci-dessous :

	probability	country	Rank
14	0.992727	[Czech RepublicCzechia]	1.0
22	0.985029	[Finland]	2.0
12	0.942216	[Spain]	3.0
3	0.935893	[France]	4.0
0	0.913119	[Ukraine]	5.0
9	0.864636	[Italy]	6.0
13	0.680652	[United KingdomUK]	7.0
15	0.674245	[Germany]	8.0
18	0.586356	[Romania]	9.0
5	0.548448	[Poland]	10.0
11	0.499459	[Serbia]	11.0
21	0.499062	[Estonia]	12.0

Figure 25 : Prédiction des probabilités avec la régression logistique

Conclusion

Il en ressort que des trois algorithmes de machine learning utilisés, seule la régression logistique me permet de prédire des potentiels gagnants. De plus, il offre une bonne précision de 64% comparativement aux autres modèles. Ce modèle sera utilisé dans la phase suivante qui est celle du déploiement.

Chapitre 6 : Déploiement

Introduction

C'est la dernière étape du processus. Il s'agit de mettre le modèle résultant en production pour l'utilisateur final. L'objectif est de mettre les connaissances acquises grâce à la modélisation dans le processus de prise de décision sous une forme appropriée.

Par conséquent, selon l'objectif, le déploiement peut aller de la simple génération d'un rapport décrivant les connaissances acquises à la mise en œuvre d'une application, permettant l'utilisation du modèle acquis pour prédire la valeur inconnue d'intérêt de l'élément.

I. Processus de déploiement

Pour déployer ma solution, on sert des outils suivants :

- Github : il stocke le code de l'application, le code de la modélisation, le modèle enregistré, les données et un fichier requirements.txt qui contient toutes les librairies dont l'application a besoin pour fonctionner ;
- Streamlit Cloud : il construit et déploie l'application web à partir du code stocké sur Github et héberge la solution sur son serveur.

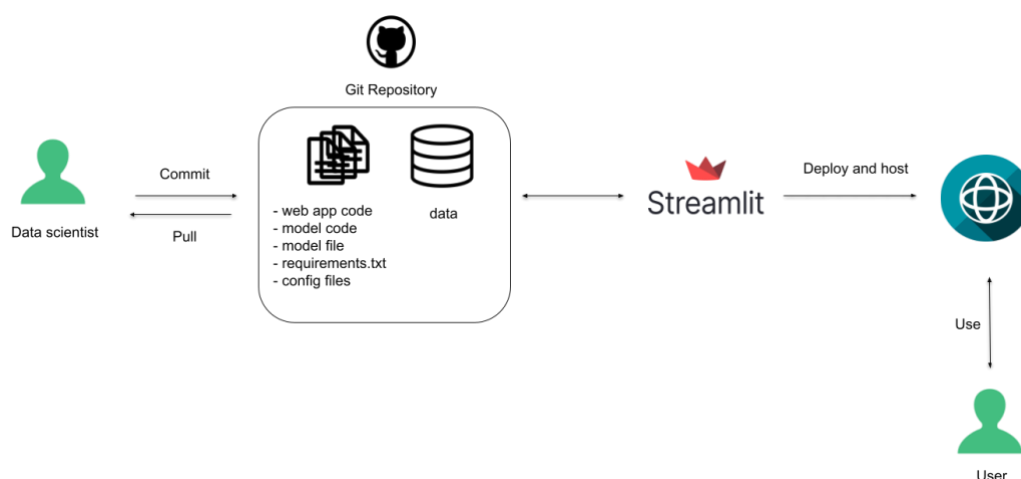


Figure 26 : Modèle de déploiement

Le choix de Streamlit comme outil de visualisation a été fait car :

- C'est une librairie python ;
- La prise en main est rapide et simple ;
- Elle propose une solution gratuite de déploiement et d'hébergement de l'application construite qui est rapide et simple ;
- Elle se met à jour automatiquement lorsque le code est mis à jour sur le répertoire Git auquel il est lié.

Le seul inconvénient constaté sur l'outil est qu'il ne permet d'énormément personnaliser le site web qu'on construit et on est limité aux composants web que propose ce dernier. Cependant ceci n'a pas été un souci pour moi car cela correspondait à mon besoin actuel.

II. Développement de l'application web

Le développement de l'application Web s'est fait de manière parallèle avec la préparation et la modélisation des données. Ceci n'a été possible que parce que le code de l'application et celui de la modélisation sont séparés. Ainsi, un problème dans la modélisation des données n'affectera pas forcément l'application web et vice versa. Les mises à jour aussi au niveau du modèle ne nécessitera pas la mise à jour de l'application web.

Ceci est d'autant plus avantageux pour moi car la collecte des données et certaines phases initiales du nettoyage des données comme la traduction prennent beaucoup de temps.

Ainsi, pour fonctionner l'application web a besoin de trois éléments essentiels au préalable :

- Les données préparées (après analyse, feature engineering ...)
- Le modèle sauvegardé au format .joblib
- Une fonction de prétraitement de données qui prépare les données à être passées dans le modèle et une fonction de prédictions qui utilise le modèle et sort les prédictions.

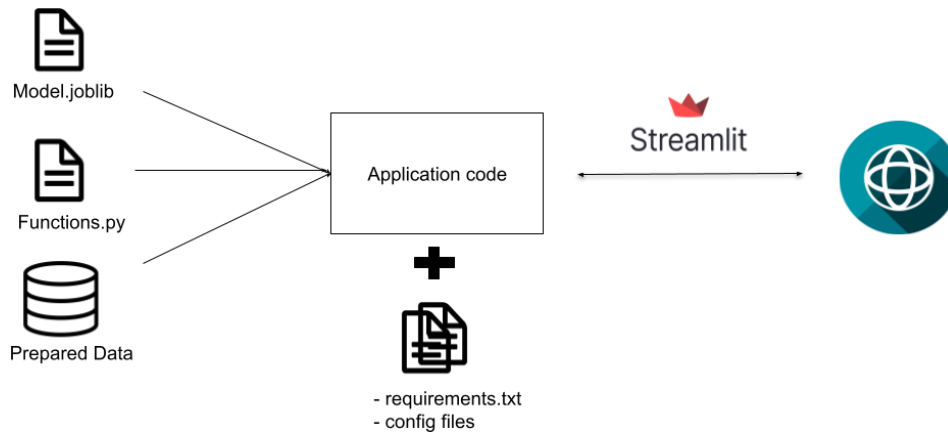


Figure 27 : Fonctionnement de l'application Web

III. Présentation de l'application WEB

L'application est organisée en deux vues principales à savoir une vue d'accueil et une vue de prédiction.

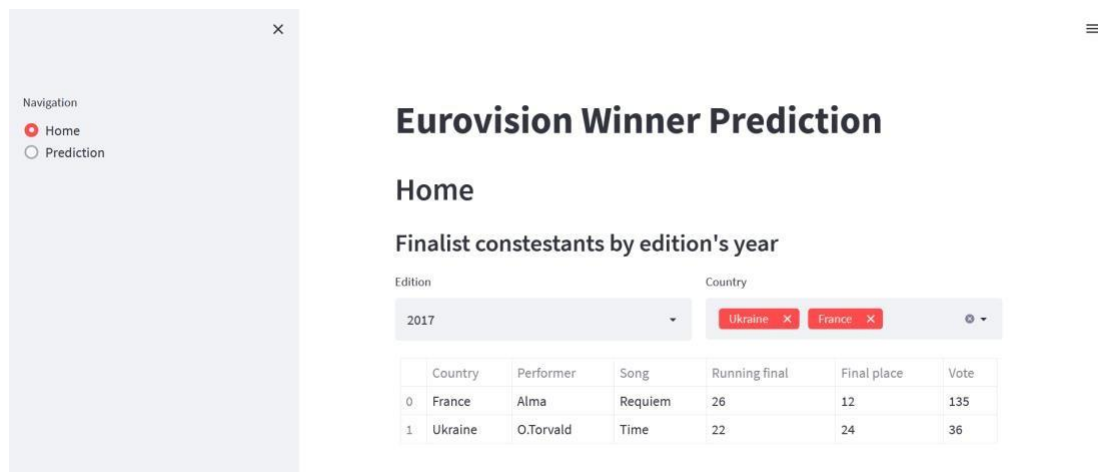


Figure 28 : Page d'accueil de l'application web

1. Page d'accueil

La vue d'accueil présente des informations générales sur les participants de l'Eurovision à savoir :

- Une liste des finalistes de l'Eurovision filtrable par édition et par pays

Finalist contestants by edition's year

Edition

2017

Country

Ukraine X

France X

Bulgaria X

	Country	Performer	Song	Running final	Final place	Vote
0	Bulgaria	Kristian Kostov	Beautiful Mess	25	2	615
1	France	Alma	Requiem	26	12	135
2	Ukraine	O.Torvald	Time	22	24	36

Figure 29 : Participants finalistes par édition

- Un graphique montrant pour tous les pays gagnants, le nombre de victoires obtenus jusqu'ici

Top 5 countries by winning frequencies

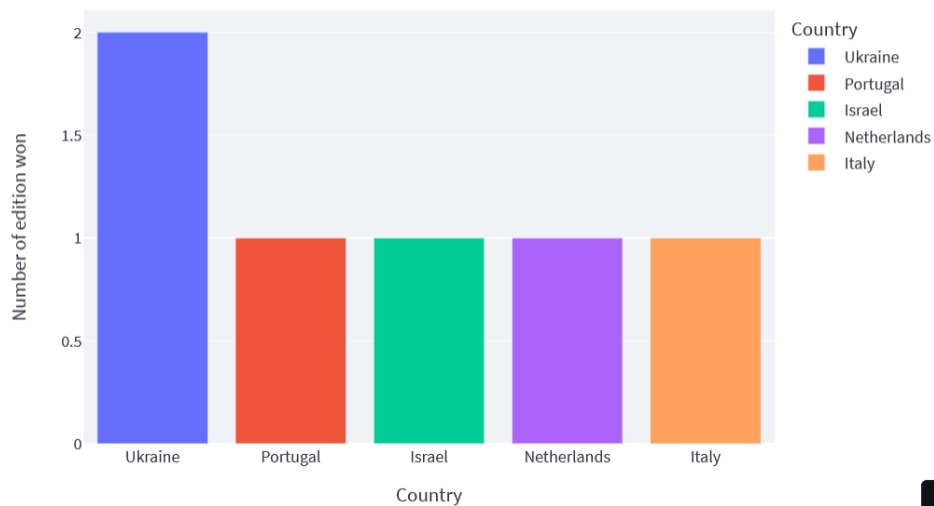


Figure 30 : Top 5 des pays ayant le plus gagné d'édition

- Un graphique montrant l'évolution des votes de la finale au cours du temps

Evolution of the point of the winner over the year

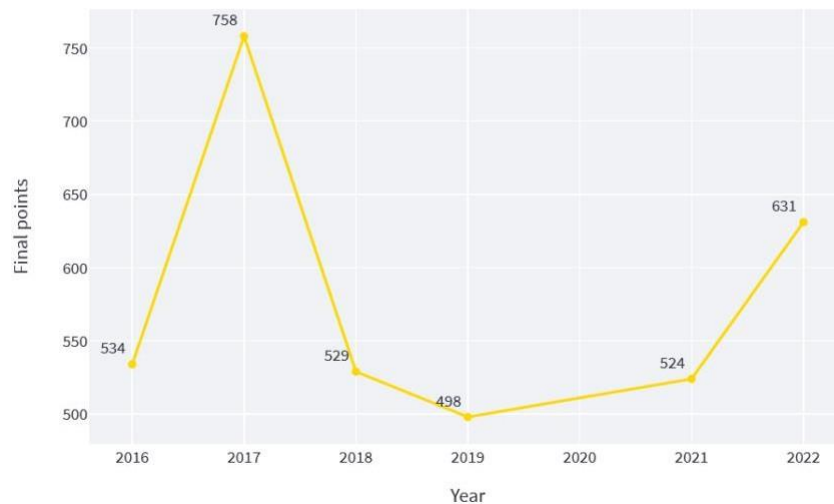


Figure 31 : Evolution des votes des gagnants au fil du temps

2. Page de prédiction

Sur la vue de prédiction, ce sont des informations obtenues suite aux traitements et à la modélisation des données initiales qu'on présente à savoir :

- L'opinion générale du public de Twitter par rapport aux pays finalistes en fonction de l'année d'édition et cette opinion peut être positive ou négative

General sentiment of the public on the contestants

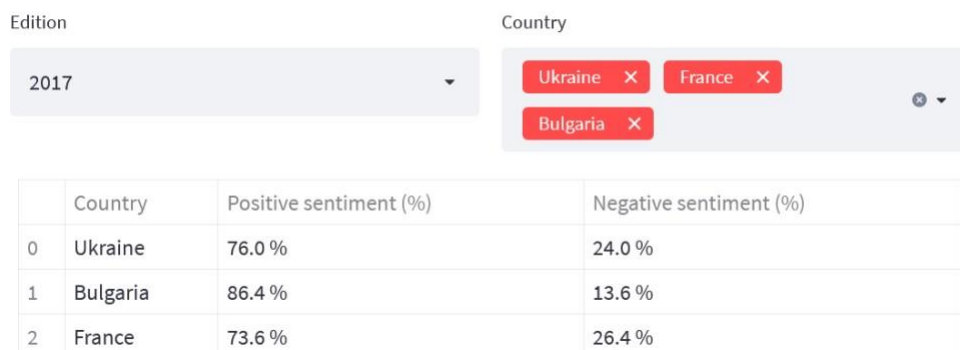


Figure 32 : Prédiction de l'opinion publique sur Twitter des participants par édition

- La prédiction du gagnant par édition

Prediction

Prediction of the winner

Edition's winner to predict

2017

Choose a country

Ukraine

	Country	Estimated rank	Winning probability
0	Ukraine	1	9.1300

Figure 33 : Prédiction des probabilités de victoire par édition et par pays

Conclusion

J'ai finalement terminé mon projet avec cette dernière étape de ma méthodologie.

Tout était inclus dans des interfaces. J'ai également réussi à tout visualiser avec l'outil Streamlit.

Conclusion et perspectives

Les réseaux sociaux sont l'un des domaines les plus intéressants sur lesquels on puisse travailler aujourd'hui. Cela est dû au fait que l'analyste travaille pour comprendre le comportement de l'utilisateur.

Ce projet a été un vrai challenge pour moi. Dans ce travail, mon intérêt s'est porté sur l'analyse des sentiments avec des techniques d'apprentissage automatique et le traitement du langage naturel dont l'objectif principal est de prédire le gagnant de l'émission EuroVision.

L'ensemble de données considéré s'appuie sur les opinions du réseau social Twitter. Un prétraitement de données a été effectué avant une visualisation et une analyse des données.

Enfin je peux conclure que ce projet a été bénéfique à plusieurs niveaux notamment l'exploration des domaines de la régression et de la classification de textes, de l'apprentissage automatique, de la visualisation de données.

Annexe 1

I. Natural language processing

Natural Language Processing (NLP) est une branche très importante du Machine Learning et donc de l'intelligence artificielle. Le NLP est la capacité d'un programme à comprendre le langage humain. Il s'agit donc d'une discipline informatique à part entière qui recouvre de nombreux sujets et méthodes, qui sont à l'origine notamment des moteurs de recherche.

La plupart des techniques de traitement naturel du langage reposent sur le Deep Learning ou apprentissage profond. Les algorithmes d'intelligence artificielle sont entraînés à partir de données afin d'apprendre à analyser le langage humain pour y trouver des patterns et des corrélations.

Les algorithmes ont pour rôle d'identifier et d'extraire les règles du langage naturel, afin de convertir les données de langage non structuré sous une forme que les ordinateurs pourront comprendre.

II. Les techniques de prétraitement

Le traitement de normalisation de texte repose principalement sur quatre tâches de base : tokenisation, normalisation de mots, suppression des stopwords et lemmatisation et stemming. Ces tâches seront détaillées dans les sections suivantes.

1. La tokenisation

La tokenisation décompose le texte brut en mots, des phrases appelées jetons. Ces jetons aident à comprendre le contexte ou à développer le modèle de la NLP. La tokenisation aide à expliquer la signification du texte en analysant la séquence des mots.

Signes de ponctuation, chiffres, liens Web et caractères spéciaux... sont généralement retirés, mais peut être mis de côté pour l'élimination. Cela dépend en grande partie du contexte et de la philosophie de l'algorithme.

2. La normalisation de mots

Normaliser le texte signifie le mettre à la même casse , généralement toutes les lettres en minuscule pour avoir les mêmes mots au début et dans toute autre partie des phrases soient représentées de la même manière.

La normalisation de mots implique également le regroupement des mots / jetons avec différentes formes dans un format standardisé, ce qui peut effectivement éliminer les traces d'erreurs d'orthographe, mais peut être utile pour obtenir des quantités standardisées et un petit nombre de jetons.

3. La lemmatisation et le stemming

Ces deux méthodes sont très couramment utilisées dans le traitement du langage naturel car permettent de représenter plusieurs dérivées d'un mot sous un même mot donc je vais uniquement garder la racine du mot.

Le processus de « lemmatisation » consiste à représenter les mots sous leur forme canonique. Par exemple pour un verbe, ce sera son infinitif. Pour un

nom, son masculin singulier. L'idée étant encore une fois de ne garder que le sens des mots utilisés dans le corpus.

Dans le cas du Stemming, consiste à ne retenir que la racine des mots étudiés. Le but étant de supprimer les suffixes, préfixes et autres des mots afin de ne conserver que leur origine. C'est un procédé plus simple que la lemmatisation et plus rapide à exécuter puisqu'on les mots sont essentiellement tronqués, contrairement à la lemmatisation qui nécessite un dictionnaire.

4. Suppression de stopwords

Vient ensuite l'étape de suppression des stopwords qui est cruciale, car elle va supprimer dans le texte tous les mots qui n'ont que peu d'intérêt sémantique. Les mots vides sont en effet tous les mots les plus courants d'une langue (déterminants, pronoms, etc..) qui n'ont aucune valeur informative pour la compréhension du sens d'un document et corpus. Ils sont très fréquents et ralentissent mon travail : je voulais donc les supprimer.

5. Autres techniques :

Le prétraitement peut inclure d'autres étapes telles que la normalisation de la casse ou la suppression des accents. Le contexte des réseaux sociaux implique également un certain nombre de traitements spécifiques, d'où on peut citer :

- Suppression des liens hypertextes : Généralement, lors de l'analyse des sentiments sur du texte, les URL ne fournissent aucune information, au contraire, les URL déforment la prédiction de la subjectivité et de la polarité d'un texte donné. En prenant l'exemple de « www.magnifique.fr », ce texte serait de classe positive, alors qu'il est

totale­ment neutre, cette mau­vaise pré­dic­tion se­rait due à la pré­sen­ce du mot opinion dans l'url.

- Supprimer les lettres en double : les internautes ont tendance à utiliser une série de lettres en double dans le même mot pour exprimer la force du sens du mot. Cependant, ce genre de répétition de lettres produira des mots erronés qui ne se trouvent pas dans le dictionnaire, c'est pourquoi les lettres obsolètes doivent être éliminées.
- Substitution de hashtags : les hashtags sont une sorte de marque sous la forme d'un ou plusieurs mots reliés entre eux et commençant par un dièse. Ce style d'écriture est souvent utilisé sur les réseaux sociaux, en taguant le contenu avec des mots-clés pour les mettre en valeur.

Annexe 2

I. Modélisation

1. Prédiction avec du Machine learning

Le machine learning ou apprentissage automatique est un domaine scientifique, et plus particulièrement une sous-catégorie de l'intelligence artificielle. Elle consiste à laisser des algorithmes découvrir des " patterns ", à savoir des motifs récurrents, dans les ensembles de données. Ces données peuvent être des chiffres, des mots, des images, des statistiques... L'apprentissage automatique utilise des méthodes statistiques et des algorithmes auto-améliorés, tout comme les humains apprennent de l'expérience passée, mais il est généralement plus efficace.

Tout ce qui peut être stocké numériquement peut être utilisé comme données pour le l'apprentissage automatique. En détectant les patterns dans ces données, les algorithmes apprennent et améliorent leurs performances lors de l'exécution de tâches spécifiques.

Parfois, interpréter des modèles ou extraire des informations à partir des données ne peut pas être une tâche évidente. À mesure que la disponibilité des ensembles de données augmente, la demande d'algorithmes d'apprentissage automatique qui gèrent ces complexités augmente également.

On a 3 catégories d'apprentissage : apprentissage supervisé, apprentissage non supervisé et apprentissage semi-supervisé.

Mais dans cette partie on s'intéresse à l'apprentissage supervisé.

Contrairement à l'apprentissage non supervisé, l'apprentissage supervisé est une tâche d'apprentissage automatique qui apprend des fonctions de prédiction à partir d'exemples annotés.

Cette catégorie a tendance à utiliser un ensemble de données pour générer un modèle qui utilise un vecteur de caractéristiques x pour entrer et sortir des informations pour obtenir l'étiquette correspondante du vecteur de caractéristiques.

Mon entrée n'est donc plus l'entrée de la méthode de résolution classique que j'ai définie dans le passé. Même la valeur de sortie réelle est considérée ici comme « l'entrée » de mon modèle.

to

Les applications d'apprentissage supervisé sont généralement divisées en deux catégories, la classification et la régression. Par exemple, lorsque la valeur de sortie est une catégorie telle que le genre, la couleur, la marque etc... qui est vraie ou fausse, la classification est pertinente. Lorsque la sortie est une valeur calculée réelle (comme le poids, le prix, le salaire), des problèmes de régression se produisent.

Les algorithmes supervisés, avec des balises appliquées, incluent la régression linéaire, la régression logistique, réseaux de neurones, SVM, arbre de décision, Naïve Bayes, forêts aléatoires, etc.

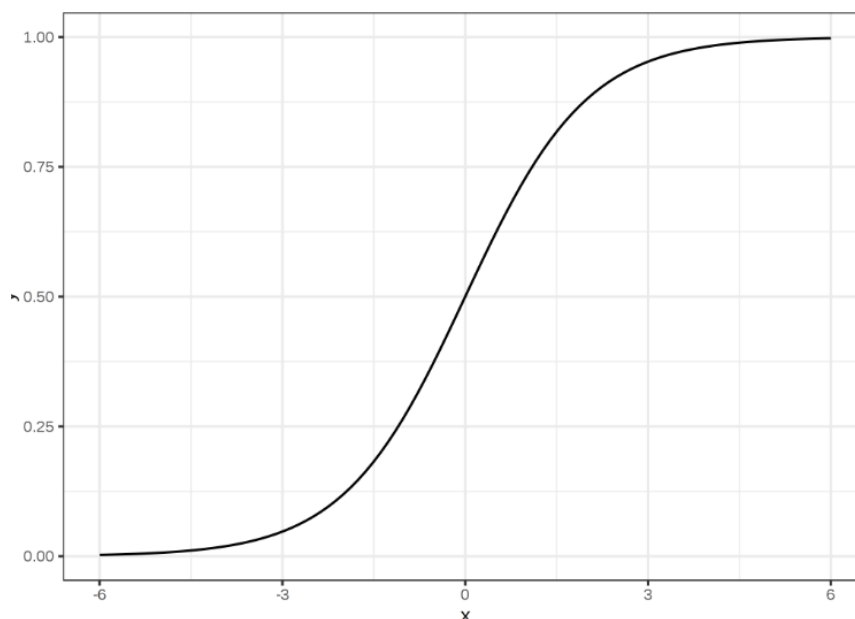
2. Techniques ML

Random Forest

Les forêts d'arbres décisionnels ou forêts aléatoires (Random Forest) sont une technique d'apprentissage d'ensemble qui s'appuie sur des arbres de décision. Le modèle de forêt aléatoire implique la création de plusieurs arbres de décision en utilisant un ensemble de données séparés à partir des données d'origine. Et en sélectionnant au hasard un sous-ensemble de variables à chaque étape de l'arbre de décision. Ensuite, le modèle sélectionne tous les modes prédits pour chaque arbre de décision.

Régression logistique

La régression logistique est similaire à la régression linéaire, mais elle est utilisée pour modéliser la probabilité d'un nombre limité de résultats (généralement deux). Il y a plusieurs raisons d'utiliser la régression logistique au lieu de la régression linéaire lors de la modélisation des probabilités de résultats. Une équation logistique est créée de telle sorte que les valeurs des résultats ne peuvent être qu'entre 0 et 1 (voir ci-dessous).



One class SVM

Le principe des SVM consiste à ramener un problème de classification ou de discrimination à un hyperplan (feature space) dans lequel les données sont séparées en plusieurs classes dont la frontière est la plus éloignée possible des points de données (ou "marge maximale"). D'où l'autre nom attribué aux SVM : les séparateurs à vaste marge. Le concept de frontière implique que les données soient linéairement séparables. Pour y parvenir, les support vector machines font appel à des noyaux, c'est-à-dire des fonctions mathématiques permettant de projeter et séparer les données dans l'espace vectoriel, les "vecteurs de support" étant les données les plus proches de la frontière. C'est la frontière la plus éloignée de tous les points d'entraînement qui est optimale, et qui présente donc la meilleure capacité de généralisation.

