**Projet Supply Chain**

Prédiction d’une note à partir d’un avis utilisateur

ligne horizontale

# 

# Introduction

## Objectif

Trustpilot a été fondée en 2007 dans l'objectif de devenir un symbole universel de confiance. Afin de rapprocher les entreprises des consommateurs afin de promouvoir la confiance et de stimuler la collaboration, Trustpilot met à disposition de ces derniers, une plateforme en ligne qui est ouverte, transparente et gratuite pour tous. Les avis Trustpilot aident les consommateurs à acheter en toute confiance et fournissent aux entreprises des données essentielles pour améliorer l'expérience client. Plus les gens utilisent notre plateforme et partagent leurs opinions, plus les informations offertes aux entreprises sont précieuses, ce qui leur permet de gagner la confiance des consommateurs du monde entier.

L’objectif premier de ce projet est d’exploiter l'ensemble des données accessibles sur les avis laissés par les utilisateurs afin de prédire la note attribuée. Nous avons choisi de nous focaliser sur un service particulier, celui de la vente de pneus à des particuliers. Un objectif optionnel sera exploré visant à identifier automatiquement les raisons d’une note basse (problème de livraison, de produit ou de relation avec les garages mandatés). Nous utiliserons pour cela des données récupérées sur le site Trustpilot : [https://fr.trustpilot.com](https://fr.trustpilot.com/)

## Contexte

Bien que nous sommes tous des consommateurs sensibles à la réputation d’une marque ou d’une enseigne, nous ne sommes pas des utilisateurs aguerris de ce type plateforme. Ainsi, l’utilisation et la manipulation de données issues de ce type de site constitue à la fois une expérience nouvelle et originale. De plus, seul Yannick RICO, en tant que chef de projet logistique et transport chez Allopneus.com, dispose d’une expérience significative dans la vente de pneus en ligne à des particuliers. Ses connaissances permettront au reste de l’équipe de mieux comprendre les enjeux auxquels est confronté la vente en ligne des pneus que ce soit sur les sources de satisfaction comme d’insatisfaction mais aussi en termes de saisonnalité.

Concernant les autres membres de l’équipe, Joan ASSELOT est ingénieur en qualité logicielle et Mickael GALLIEZ est auditeur interne au sein d’une banque d’investissement.

## Lien vers le GitHub où sont stockés les données et codes utilisés :

<https://github.com/Mickael-Galliez/mai23_cds_supply_chain.git>

# Présentation des données

## Présentation générale

La première étape du projet consiste à collecter les données depuis le site Trustpilot. Dans un premier temps, nous avons sélectionnés les 3 plus importantes sociétés référencées sur Trustpilot en nombres d’avis publiés et vérifiés: CentralePneus.fr, Allopneus et Pneus Online.

Dans un deuxième temps, nous nous sommes appuyés sur la technique du web scraping pour collecter l’ensemble des données disponibles et relatives à chacun des avis. Le web scraping est une technique permettant d’automatiser des processus de collecte de données sur le web à l’aide de robots ou de scripts automatisés appelés web crawlers.

Concernant la librairie, nous avons choisis BeautifulSoup plutôt que Selenium en raison de sa simplicité d’utilisation. BeautifulSoup fournit des méthodes simples pour naviguer, rechercher et modifier un arbre d’analyse dans des fichiers HTML ou XML. Cet outil permet non seulement de scraper, mais aussi de nettoyer les données. BeautifulSoup prend en charge l’analyseur HTML inclus dans la bibliothèque standard de Python, mais aussi plusieurs analyseurs Python tiers comme lxml ou html5lib.

Un avis client se présente de la manière suivante :



1. **Client**, nom de l’utilisateur auteur de l'avis.
2. **Nombre\_avis\_publie**, nombre d’avis soumis par l’utilisateur.
3. **Pays**, origine de l’utilisateur.
4. **Note**, sur 5 attribué par l’utilisateur.
5. **Statut**, de l’utilisateur. Le statut “vérifié” signifie que l’avis de cet utilisateur a été sollicité par le site Trustpilot à la suite de l'achat d’un service. Nous pouvons donc être très confiant sur l'honnêteté de cet avis. Si le symbole “avis vérifié” est absent, cela signifie que l’avis n’a pas fait l’objet d’une sollicitation et est donc moins fiable.
6. **Date\_publication**, date de publication de l’avis.
7. **Titre**, de l’avis rédigé par l’utilisateur.
8. **Commentaire**, commentaire de l’avis rédigé par l’utilisateur.
9. **Date\_experience**, date de l'expérience utilisateur.

Ces données correspondent aux colonnes de notre jeu de données auxquels il faut ajouter les données suivantes :

1. **Entreprise**, le nom de la société concernée (n’apparaissant pas sur l’avis mais sur la page de l'avis).
2. **Date\_reponse**, date de la réponse de la société lorsque celle-ci existe.
3. **Reponse,** le commentaire de la réponse de la société lorsque celle-ci existe.

Une fois l'opération de web scraping terminée, nous avons obtenu un jeu de données composé de 12 colonnes et 286 000 entrées.

## Variable explicative

À la suite d’une première analyse des données collectées, la variable explicative qui semble la plus appropriée pour déterminer la note attribuée par un utilisateur est **Commentaire**, c’est-à-dire le commentaire de l’avis rédigé par l’utilisateur.

## Variable cible

L’objectif de ce projet étant de prédire la note attribuée en fonction des variables explicatives ci-dessus, la variable cible adéquate est **Note**. Dans le jeu de données initial, elle prend cinq valeurs différentes : ['1', '2', '3', '4', '5']

# Pré-traitement des données

L’objectif de ce projet étant de prédire une note en fonction d’un commentaire laissé par un client sur Trustpilot, il est nécessaire de disposer d’une quantité suffisamment importante de donnée pour faire tourner correctement notre modèle et l’entraîner. Cependant, disposer d’un jeu de données significatif ne suffit pas pour obtenir un algorithme performant car les données dont nous disposons ne sont pas systématiquement adaptées et il faut la plupart du temps les traiter préalablement pour pouvoir ensuite les utiliser.

En effet, des erreurs d’acquisition, liées à un mauvais paramétrage de notre code ou à une mauvaise exécution de notre scripte, peuvent corrompre notre jeu de données et biaiser l’entraînement. Parmi ces erreurs, nous pouvons citer des informations incomplètes, des valeurs manquantes ou erronées. Il est donc souvent indispensable d’établir une stratégie de prétraitement des données à partir de nos données brutes pour arriver à des données exploitables et qui nous donneront un modèle plus performant.

## Consolidation des données

Le scripte de web scraping que nous avons construit à partir de BeautifulSoup a été appliqué sur chacune des entreprises sélectionnées. Une fois ces données collectées, elles ont été intégrées dans 3 DataFrame distinct et exporter dans des fichiers Excel. Cependant, il est nécessaire de consolider nos données en un seul fichier avant de poursuivre notre pré-traitement des données. Pour cela, il est nécessaire de s’appuyer sur la libraire Pandas et OS afin de créer une boucle.

* Création d’une liste pour stocker les DataFrames

liste\_fichiers = []

* Création d'une boucle permettant la récupération des fichiers

for fichier in os.listdir(r'.\Data'):

if fichier.endswith('.xlsx'):

chemin\_fichier = os.path.join(r'C:\Data ', fichier)

df = pd.read\_excel(chemin\_fichier)

liste\_fichiers.append(df)

* Concaténation de tous les fichiers en un seul fichier

df\_complet = pd.concat(liste\_fichiers)

Voici les 5 premières lignes de notre DataFrame issues de notre fichier consolidé :

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

* Renommer les colonnes afin de remplacer les espaces et supprimer les accents

nom\_colonnes = {'Nombre d’avis publié': 'Nombre\_avis\_publie', 'Date de l’expérience': 'Date-experience', 'Date de publication': 'Date\_publication', 'Date de réponse': 'Date\_reponse'}

df.rename(columns = nom\_colonnes, inplace=True)

## Nettoyage des données

Une fois nos données consolidées, il est important d’identifier puis de supprimer les valeurs manquantes ou erronées présentes dans notre variable explicative **Commentaire** comme dans notre variable cible **Note**.

* Suppression des lignes vides dans la colonne **Commentaire**

nombre\_de\_lignes\_vides = df\_complet['Commentaire'].isnull().sum()

df\_complet.dropna(subset=['Commentaire'], inplace=True)

Nombre de lignes à supprimer : 16504

* Suppression des valeurs autres que celles intégrant des notes dans la colonne **Note**

valeurs\_valides = ["Noté 5 sur 5 étoiles", "Noté 4 sur 5 étoiles", "Noté 3 sur 5 étoiles", "Noté 2 sur 5 étoiles", "Noté 1 sur 5 étoiles"]

lignes\_a\_supprimer = df\_complet[~df\_complet ['Note'].isin(valeurs\_valides)]

df.drop(lignes\_a\_supprimer.index, inplace=True)

Nombre de lignes à supprimer : 3568

* Suppression des doublons

df\_complet.drop\_duplicates(inplace=True)

Nombre de lignes à supprimer : 23

## Transformations des données

Comme nous pouvons le voir dans l’image ci-dessus, certaines colonnes contiennent à la fois des données texte et des données numériques. C’est notamment le cas pour les colonnes **Nombre\_avis\_publie** et **Note,** dont on souhaite uniquement garder les valeurs numériques.

* Transformation des valeurs de ces colonnesen données numériques

df\_complet["Nombre\_avis\_publie"] = df\_complet["Nombre\_avis\_publie"].str.replace(' avis', '')

df\_complet["Note"] = df\_complet["Note"].str.extract(r'(\d+)')

En observant notre DataFrame, on peut s’apercevoir qu’on a une problématique similaire au niveau de **Date\_publication**, **Date\_experience** et **Date\_reponse**. Pour transformer ces colonnes en colonne avec uniquement des données de date, il faut s’appuyer, en plus de Pandas, sur la librairie datetime.

* Conversion des valeurs de ces colonnes en données date et temps

locale.setlocale(locale.LC\_TIME, 'fr\_FR.UTF-8')

df\_complet["Date\_experience"] = df\_complet["Date\_experience"].str.replace("Date de l'expérience: ", "")

locale.setlocale(locale.LC\_TIME, '')

df\_complet["Date\_experience"]= pd.to\_datetime(df\_complet["Date\_experience"], format='%d %B %Y').dt.strftime('%Y-%m-%d')

df\_complet["Date\_publication"] = pd.to\_datetime(df\_complet["Date\_publication"], format='%Y-%m-%dT%H:%M:%S.%fZ').dt.date

df\_complet["Date\_publication"] = pd.to\_datetime(df\_complet["Date\_publication"], errors='coerce')

df\_complet["Date\_reponse"] = pd.to\_datetime(df\_complet["Date\_reponse"], format='%Y-%m-%dT%H:%M:%S.%fZ').dt.date

# Visualisation des données

Dans le cadre de notre projet, la visualisation des données, c’est-à-dire, le fait de représenter visuellement ses données, est importante pour pouvoir déceler et comprendre les informations que l’on a collecté car les données brutes sont difficilement interprétables et exploitables. Ce processus se réalise via des librairies dédiées tel que Matplotlib et Seaborn.

* Matplotlib est une bibliothèque conçue pour tracer et visualiser des graphiques via le langage de programmation Python. Matplotlib offre de nombreuses fonctionnalités pour personnaliser les graphes. Les utilisateurs peuvent contrôler presque tous les aspects du tracé, y compris la taille du graphique, la résolution, les couleurs, les styles de ligne, les marqueurs, les fonds, les échelles, les textes, les polices de caractères, etc.
* Seaborn est une bibliothèque permettant de créer des graphiques statistiques en Python. Elle est basée sur Matplotlib, et s’intègre avec les structures Pandas. C’est une bibliothèque aussi performante que Matplotlib, mais qui apporte des fonctionnalités supplémentaires facilitant l’exploration et la compréhension des données.

***Répartition des avis collectés par entreprise via un graphique de type Pie chart***

Une image contenant texte, capture d’écran, diagramme, cercle

Description générée automatiquement

*Interprétation* : la majorité des avis collectées (environ 2/3) proviennent d’une même entreprise. Ici, CentralePneus.fr.

***Evolution de la moyenne des notes via un graphique de type Line chart***

Une image contenant texte, ligne, diagramme, Tracé

Description générée automatiquement

*Interprétation* : contrairement à ses concurrents qui ont connu une relative stabilité de leurs notes, la moyenne des notes d’Allopneus a souffert d’une baisse importante en 2013 et 2016.

***Répartition des notes par entreprise via un graphique de type Bar chart***

Une image contenant texte, capture d’écran, diagramme, Tracé

Description générée automatiquement

*Interprétation* : ce graphique présente la répartition des notes par entreprises. Il permet de visualiser rapidement la prépondérance d’une note par rapport aux autres.

***Répartition des réponses par entreprise via un graphique de type Violin chart***

Une image contenant diagramme, Tracé, origami

Description générée automatiquement

*Interprétation* : on s’aperçoit que contrairement à Pneus Online, les deux autres entreprises répondent uniquement aux avis ayant des notes comprises entre 1 et 3.

***Répartition des réponses par entreprise via un graphique de type Violin chart***

Une image contenant texte, capture d’écran, ligne, Tracé

Description générée automatiquement

*Interprétation* : on s’aperçoit que contrairement à Pneus Online, les deux autres entreprises répondent uniquement aux avis ayant des notes comprises entre 1 et 3.

***Corrélation entre le temps moyen de réponse et la note via un graphique de type Heat Map***

Une image contenant texte, capture d’écran, Police, nombre

Description générée automatiquement

*Interprétation* : l’objectif de ce graphique est d’établir une corrélation entre le temps moyen de réponse et la note étudié. Ainsi, les

nombre mesurant la dépendance entre deux caractères quantitatifs.