

Laboratoire: Amazon SageMaker, introduction a jupyter NoteBooks.

Scénario: amélioration des performances des Quiz a CloudyCoding College.

CloudyCoding College, un établissement fictif spécialisé dans l'enseignement du développement logiciel et de la programmation, a récemment introduit des quiz réguliers pour évaluer la rétention des connaissances de ses étudiants. Ces quiz sont essentiels pour identifier les lacunes des étudiants et améliorer le programme éducatif. Cependant, lors d'une récente évaluation, les résultats des étudiants étaient nettement inférieurs aux attentes, malgré un impact uniforme sur l'ensemble de la classe.

L'équipe pédagogique soupçonne qu'il existe des problèmes dans la formulation des questions, la structure des quiz, ou peut-être un manque d'engagement des étudiants.

Problème à Résoudre

L'objectif est d'identifier les causes potentielles des faibles performances des étudiants aux quiz et d'apporter des améliorations ciblées aux méthodes pédagogiques. Cela inclut :

- L'analyse des tentatives de quiz des étudiants pour identifier les questions qui posent problème.
- La compréhension des tendances dans les réponses des étudiants pour ajuster les cours et fournir des ressources supplémentaires.
- L'optimisation de la conception des quiz pour améliorer l'engagement et la rétention des connaissances.

Solution Proposée : Utilisation d'Amazon SageMaker avec Jupyter Notebooks

Pour résoudre ce problème de **CloudyCoding College** vous en tant que spécialiste en AWS Machine Learning vous allez proposer d'utiliser **Amazon SageMaker** et **Jupyter Notebooks** pour analyser les données des tentatives de quiz. Voici les étapes de votre solution :

1. **Collecte et Chargement des Données** : les données des tentatives de quiz (questionnaires, réponses, résultats des étudiants) sont déjà stockées sous forme de fichiers CSV. Le département informatique mis cela a votre disposition et vous allez l'importer dans un **Jupyter Notebook** pour les traiter et les analyser.
2. **Analyse des Données** :
 - En utilisant **pandas** pour charger et manipuler les données, vous allez analyser le nombre de tentatives par question (`dataset["questionId"].value_counts()`).
 - Cela permettra d'identifier quelles questions ont le taux de réussite le plus bas, indiquant des problèmes potentiels dans leur formulation ou leur difficulté.
3. **Visualisation des Résultats** :
 - Grâce à **matplotlib**, les données seront visualisées sous forme de graphiques pour montrer les tendances des performances des étudiants.
 - Des graphiques montrant la distribution des scores des étudiants, les taux de réussite par question, et les performances générales seront générés.
4. **Identification des Questions Problématiques** :
 - Les questions qui ont un taux de réussite très faible seront identifiées, et l'équipe pédagogique les réexaminera pour s'assurer qu'elles sont bien formulées et alignées sur les objectifs d'apprentissage.
 - Une analyse supplémentaire pourrait identifier si certains sujets sont moins bien compris par les étudiants, en fonction des performances sur des catégories de questions spécifiques.
5. **Proposition d'Améliorations** :
 - En fonction des analyses, des recommandations seront faites à l'équipe pédagogique pour réviser les questions problématiques.
 - L'analyse pourrait également révéler des motifs où certaines périodes d'enseignement (ex : avant les examens) ou certains jours de la semaine ont un impact sur les résultats des étudiants, menant à des ajustements du calendrier d'évaluation.

Description

Les carnets Jupyter ou Jupyter Notebook sont un outil puissant utilisé principalement dans de nombreux projets de science des données et d'apprentissage automatique pour permettre à un individu ou à un groupe de construire, documenter et visualiser leur code de manière collaborative dans un environnement interactif. Cette solution open source peut être auto-hébergée, et est également disponible en tant que solution gérée sur la plupart des plateformes cloud majeures.

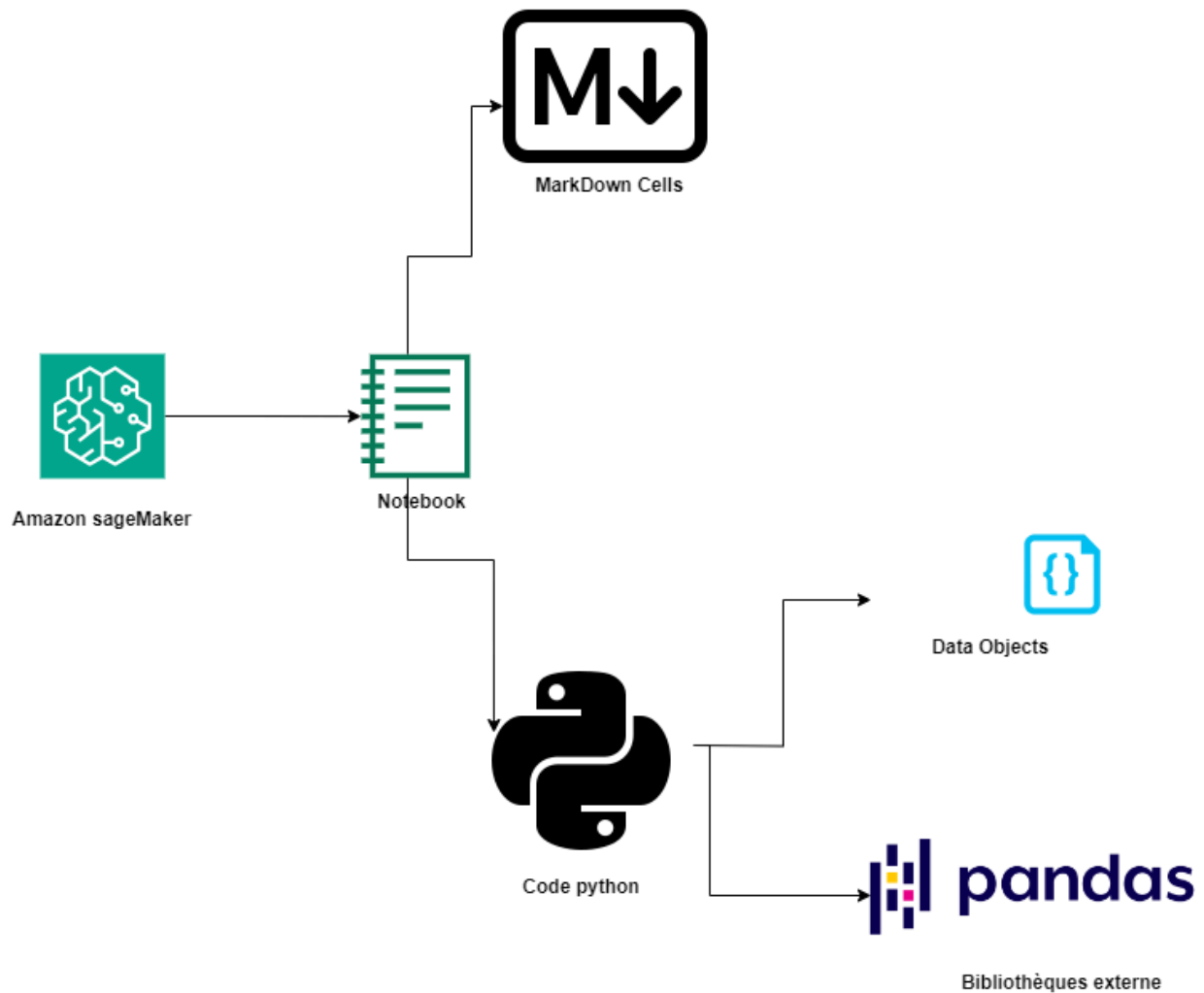
Dans ce laboratoire, nous allons utiliser un **Notebook Jupyter** avec Amazon **SageMaker** pour utiliser un notebook existant, exécuter du code existant écrit par d'autres, et aussi écrire et exécuter notre propre code. Des connaissances de base en Python seront utiles, mais ne sont pas nécessaires.

Coûts: offre gratuit, soit **250 heures** d'utilisation des instances **ml.t3.medium** ou **ml.t2.medium** sur les **instance de Notebook**

Temps estimé de réalisation: 1h30 minutes

Architecture:

Introduction a Jupyter Notebooks avec Amazon SageMaker



Objectifs

Réussir ce laboratoire en atteignant les objectifs d'apprentissage suivants :

Etape 1: Ouvrir le Notebook Jupyter existant

- Naviguez pour ouvrir l'instance Amazon SageMaker Notebook existante, et lancez le fichier Jupyter Notebook fourni.
- Dans la barre de recherche supérieure, tapez « Sagemaker » pour rechercher le service Sagemaker.

- Cliquez sur le résultat Amazon SageMaker pour accéder directement au service Sagemaker.
- Cliquez sur le bouton Notebook Instances pour consulter le carnet de notes fourni par le laboratoire.

▼ Applications and IDEs

Studio

Canvas

RStudio

TensorBoard

Profiler

Notebooks ←

- Cliquez sur create Notebook instance

The screenshot shows the Amazon SageMaker console interface. At the top, there are two tabs: 'Notebook instances' (selected) and 'Git repositories'. Below the tabs, the 'Notebook instances' section is displayed. It includes a search bar with the placeholder text 'Search notebook instances', a refresh button, an 'Actions' dropdown menu, and a prominent orange 'Create notebook instance' button. A red checkmark is drawn over the 'Create notebook instance' button. Below the search bar, there is a table with columns: 'Name', 'Instance', 'Creation time', 'Status', and 'Actions'. The table is currently empty, with the message 'There are currently no resources.' displayed below it.

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

Notebook instance settings

Notebook instance name

MyNotebook-instance

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.t3.medium

Platform identifier [Learn more](#)

Amazon Linux 2, Jupyter Lab 3

► Additional configuration

Root access - optional

- ☒ Enable - Give users root access to the notebook
☐ Disable - Don't give users root access to the notebook
Lifecycle configurations always have root access

Encryption key - optional

Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.

No Custom Encryption

► Network - optional

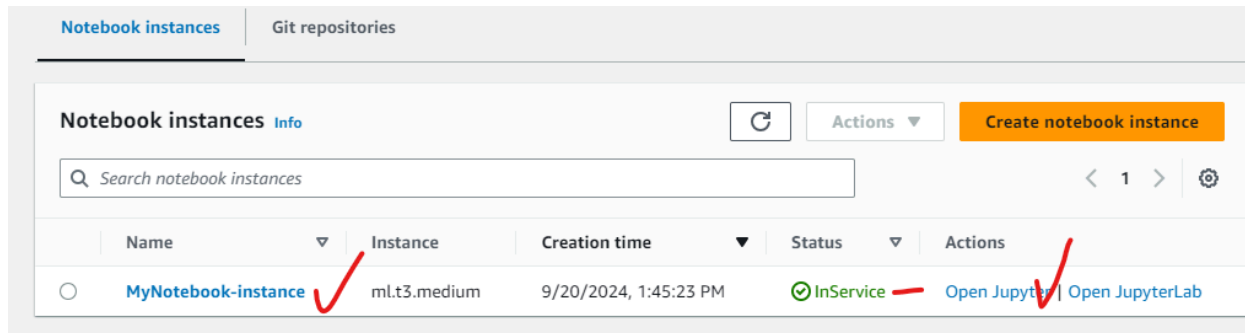
► Git repositories - optional

► Tags - optional

Cancel

Create notebook instance

- Patientons que le serveur s'initialise, une fois que le serveur est ok, nous allons visualiser le contenu de notre notebook instance.



- Normalement notre instance NoteBook sera vide, donc c'est à nous d'uploader les données ou fichiers à l'intérieur comme sur l'image ci-dessous:



NB: il faut uploader les données qui ont été fournies pour ce laboratoire, uploader tout le contenu du dossier.



Etape 2: Exécuter le code de démonstration

- Avant d'entrer dans notre scénario, nous allons effectuer une démonstration de base du fonctionnement des Notebooks Jupyter et de la manière dont ils peuvent être utiles pour nos objectifs.

Etape 3: Effectuer une analyse de données de base

- Importez le jeu de données nécessaire des résultats de notre quiz en utilisant pandas, et exécutez l'analyse de base pour confirmer à la fois le nombre de questions dans notre jeu de données, et combien de tentatives ont été faites pour chaque question.

Assessment Review Notebook

CloudyCoding College is a fictional educational institution that uses regular short quizzes to test their students knowledge retention, and identify or for improvement. This notebook has been created to provide a platform to generate insights into recent quiz attempts.

This notebook provides access to the most recent quiz attempt by 100 of the college's students for analysis and quality improvement. All data has completely de-identified.

Demo Orientation

Before we get started, let's take a look at how code is actually executed in a Jupyter Notebook.

We'll start with a basic calculation - what do you think it will do? Run the cell when you're ready.

```
In [9]: a = 5
        b = 10
        print(a + b)
```

15

- Exécutez la deuxième cellule de code pour faire la même chose, mais avec une multiplication utilisant les variables assignées précédemment.

AssessmentReviewNotebook Last Checkpoint: 21 minutes ago (unsaved changes)

conda_python3

```
print(a + b)
```

15

One of the most powerful parts of a Jupyter Notebook is that it uses a continuous interactive session of Python, so code run in one cell can be used in other completely different cell, as long as it's already been run.

```
In [10]: print(a * b)
```

50

- Exécutez la troisième cellule de code pour réaffecter les deux variables précédemment affectées.
- Exécutez à nouveau la deuxième cellule de code pour constater que le résultat a changé, puisque les variables ont changé.

```
print(a + b)
15

One of the most powerful parts of a Jupyter Notebook is that it uses a continuous interactive session of Python, so code run in one cell can be used in other completely different cell, as long as it's already been run.

In [10]: print(a * b)
50

We can also run cells in a different order, making it easier to experiment with changes to our code. Let's run the cell below, then run the multiplication in the cell above. Despite this cell not doing anything on its own, it does change how the multiplication is executed, since we've changed the underlying variables.

In [11]: a = 50
b = 3.5
```

- Exécutez la quatrième cellule de code pour importer la bibliothèque random et définir la fonction is_odd_or_even, que nous utiliserons dans la cellule suivante.

```
In [5]: import random

def is_odd_or_even(i):
    if (i % 2 == 1):
        return str(i) + " is an ODD number"
    else:
        return str(i) + " is an EVEN number"
```

- Exécutez la cinquième cellule de code pour utiliser notre fonction précédente afin de vérifier si les variables assignées précédemment et un nombre aléatoire sont soit des nombres pairs, soit des nombres impairs.

```
In [12]: # Using one of our earlier variables
print( is_odd_or_even(a) )

# Using a random number between 1 and 1,000,000
print( is_odd_or_even(
    random.randrange(1,100000)
) )

50 is an EVEN number
94292 is an EVEN number
```

Étape 4: initialiser votre Notebook

- Dans la section Initialiser notre Notebook, exécutez la première cellule de code pour importer les paquets pandas et matplotlib nécessaires. Les deux sont installés par défaut dans notre environnement.
- Exécutez la deuxième cellule de code pour importer le CSV existant dans la variable dataset, et pour imprimer le nombre d'enregistrements (lignes) dans le CSV.

jupyter AssessmentReviewNotebook Last Checkpoint: 36 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Now, back to our scenario:

Initializing our Notebook

We need to load our dataset and initialize our Notebook, and ensure Python is working as expected.

```
In [13]: import pandas as pd
import matplotlib.pyplot as plt

In [8]: dataset = pd.read_csv('question-data.csv')
print(len(dataset))
1000
```

- Exécutez la troisième cellule pour imprimer les premières lignes du tableau.

jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Let's take a look at the dataset and see how it looks:

```
In [14]: dataset.head()
```

Out[14]:

	attemptId	questionId	submittedAnswer	correctAnswer	isCorrect
0	11f2b583-f138-4b22-2c98-27da86eda3f8	811fa459-5bcd-9e2e-f8d8-391c4f4507ad	d	d	True
1	f88e9c84-3b4b-d933-5a59-4fbcf65ea2a1	201922bd-bc6d-ada9-e842-643be22558f8	c	c	True
2	0197b07a-2937-343e-c116-1bb65dbd88bc	989d68ee-4636-b9f4-eb5d-543258e1315a	d	d	True
3	44ecbf04-27c8-7fa6-06a3-048dabb6dc5d	b45c6079-9179-340c-6ad1-17430933f22a	d	d	True
4	5ac46b5d-3625-6a28-ad93-f58d71721720	7b096543-245c-709e-8433-3b08affd72b7	b	b	True

jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Basic Data Analysis

Let's confirm how many unique questions are present in the dataset based on the question ID, and how many attempts there have been at each. This is where we'll start using some visualizations and leveraging the power of Jupyter Notebooks.

```
In [15]: dataset["questionId"].value_counts()
```

Out[15]:

```
questionId
811fa459-5bcd-9e2e-f8d8-391c4f4507ad    100
201922bd-bc6d-ada9-e842-643be22558f8    100
989d68ee-4636-b9f4-eb5d-543258e1315a    100
b45c6079-9179-340c-6ad1-17430933f22a    100
7b096543-245c-709e-8433-3b08affd72b7    100
b644e965-f0da-2873-fb42-10b50bd1d9ca    100
e57b8a51-d2dc-dadc-edfe-6f7f5b3caa6d    100
07223e8f-99f3-0531-1e1f-c54f6bf8f8d7    100
f668fe79-77f4-4d93-4069-2cb864a7c62e    100
b48e7446-cf17-3171-276e-c83305f3ce76    100
Name: count, dtype: int64
```

Etape 5: Approfondissement

- Dans la section Approfondir, sélectionnez la cellule de code vide, utilisez le menu déroulant de la barre d'outils pour la transformer d'une **cellule de code** en une **cellule Markdown** et ajoutez une description de ce que nous sommes sur le point de faire.

Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted | conda_pyt

Out[15]:

questionId	
811fa459-5bcd-9e2e-f8d8-391c4f4507ad	100
201922bd-bc6d-ada9-e842-643be22558f8	100
989d68ee-4636-b9f4-eb5d-543258e1315a	100
b45c6079-9179-340c-6ad1-17430933f22a	100
7b096543-245c-709e-8433-3b08affd72b7	100
b644e965-f0da-2073-fb42-10b50bd1d9ca	100
e57b8a51-d2dc-dadc-edfe-6f7f5b3caa6d	100
07223e8f-99f3-0531-1e1f-c54f6bf8f8d7	100
f668fe79-77f4-4d93-4069-2cb864a7c62e	100
b48e7446-cf17-3171-276e-c83305f3ce76	100
Name: count, dtype: int64	

Diving Deeper

We know the results from the quizzes was lower than expected, but all students were equally impacted. In this section, try and identify the root cause:

In []:

Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted | conda_python3 Logout

Out[15]:

questionId	
811fa459-5bcd-9e2e-f8d8-391c4f4507ad	100
201922bd-bc6d-ada9-e842-643be22558f8	100
989d68ee-4636-b9f4-eb5d-543258e1315a	100
b45c6079-9179-340c-6ad1-17430933f22a	100
7b096543-245c-709e-8433-3b08affd72b7	100
b644e965-f0da-2073-fb42-10b50bd1d9ca	100
e57b8a51-d2dc-dadc-edfe-6f7f5b3caa6d	100
07223e8f-99f3-0531-1e1f-c54f6bf8f8d7	100
f668fe79-77f4-4d93-4069-2cb864a7c62e	100
b48e7446-cf17-3171-276e-c83305f3ce76	100
Name: count, dtype: int64	

Diving Deeper

We know the results from the quizzes was lower than expected, but all students were equally impacted. In this section, try and identify the root cause:

- Par exemple un texte de ce genre : ”Commençons par évaluer le taux de réussite de chaque question et identifions les éventuelles valeurs aberrantes. Nous pouvons ensuite examiner ces questions aberrantes pour vérifier la qualité de la question elle-même ou déterminer si le matériel de formation original a besoin d'être amélioré.”

Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Out[15]:

```
questionId
811fa459-5bcd-9e2e-f8d8-391c4f4507ad    100
201922bd-bc6d-ada9-e842-643be22558f8    100
989d68ee-4636-b9f4-eb5d-543258e1315a    100
b45c6079-9179-340c-6ad1-17430933f22a    100
7b096543-245c-709e-8433-3b08affd72b7    100
b644e965-f0da-2073-fb42-10b50bd1d9ca    100
e57b8a51-d2dc-dadc-edfe-6f7f5b3caa6d    100
07223e8f-99f3-0531-1e1f-c54f6bf8f8d7    100
f668fe79-77f4-4d93-4069-2cb864a7c62e    100
b48e7446-cf17-3171-276e-c83305f3ce76    100
Name: count, dtype: int64
```

Diving Deeper

We know the results from the quizzes was lower than expected, but all students were equally impacted. In this section, try and identify the root cause:

Commençons par évaluer le taux de réussite de chaque question et identifions les éventuelles valeurs aberrantes. Nous pouvons ensuite examiner ces questions aberrantes pour vérifier la qualité de la question elle-même ou déterminer si le matériel de formation original a besoin d'être amélioré.

- Exécutez la cellule Markdown pour rendre le texte.

Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted conda_python3

Code

```
b45c6079-9179-340c-6ad1-17430933f22a    100
7b096543-245c-709e-8433-3b08affd72b7    100
b644e965-f0da-2073-fb42-10b50bd1d9ca    100
e57b8a51-d2dc-dadc-edfe-6f7f5b3caa6d    100
07223e8f-99f3-0531-1e1f-c54f6bf8f8d7    100
f668fe79-77f4-4d93-4069-2cb864a7c62e    100
b48e7446-cf17-3171-276e-c83305f3ce76    100
Name: count, dtype: int64
```

Diving Deeper

We know the results from the quizzes was lower than expected, but all students were equally impacted. In this section, try and identify the root cause:

Commençons par évaluer le taux de réussite de chaque question et identifions les éventuelles valeurs aberrantes. Nous pouvons ensuite examiner ces questions aberrantes pour vérifier la qualité de la question elle-même ou déterminer si le matériel de formation original a besoin d'être amélioré.

In []:

- Créez une nouvelle cellule et saisissez le code suivant pour effectuer l'analyse nécessaire :

```
success_rate = dataset.groupby('questionId')['isCorrect'].mean()
```

```
plt.bar(success_rate.index, success_rate.values)
```

```
for i, rate in enumerate(success_rate):
    plt.text(i, rate, f'{rate:.0%}', ha='center', va='bottom')
```

```
plt.xlabel('Question ID')
```

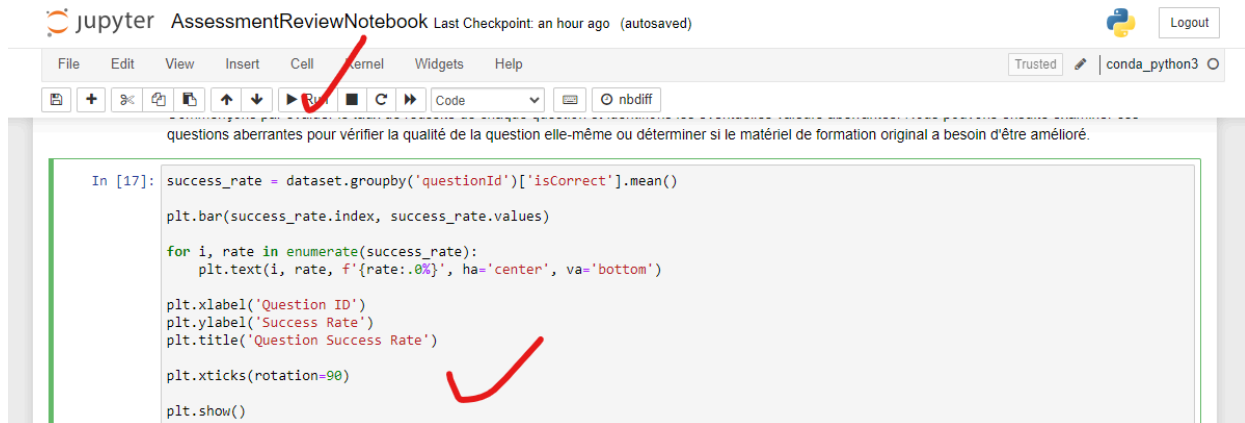
```
plt.ylabel('Success Rate')
```

```
plt.title('Question Success Rate')
```

```
plt.xticks(rotation=90)
```

```
plt.show()
```

Run the cell to perform the analysis by generating the bar chart graphic.



Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted | conda_python3

questions aberrantes pour vérifier la qualité de la question elle-même ou déterminer si le matériel de formation original a besoin d'être amélioré.

```
In [17]: success_rate = dataset.groupby('questionId')['isCorrect'].mean()

plt.bar(success_rate.index, success_rate.values)

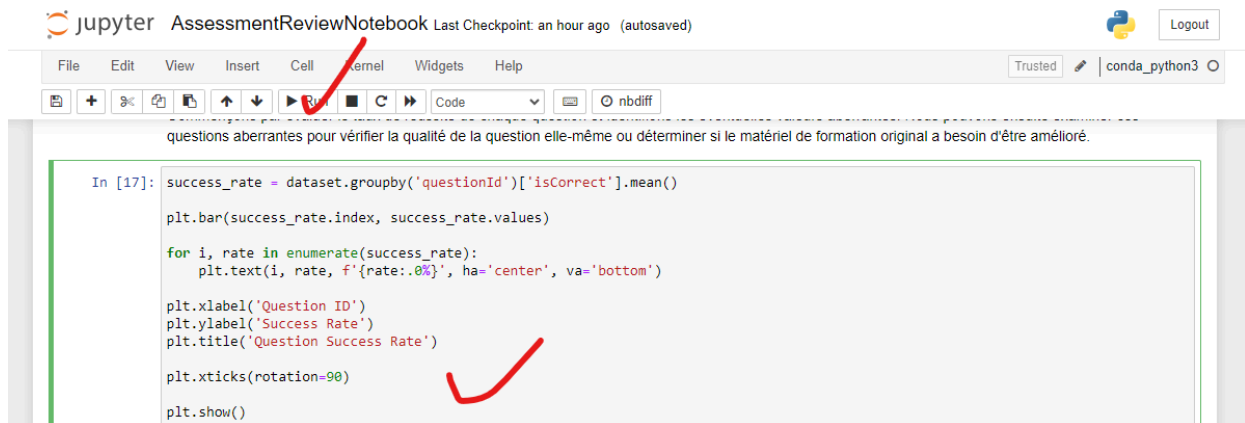
for i, rate in enumerate(success_rate):
    plt.text(i, rate, f'{rate:.0%}', ha='center', va='bottom')

plt.xlabel('Question ID')
plt.ylabel('Success Rate')
plt.title('Question Success Rate')

plt.xticks(rotation=90)

plt.show()
```

- Exécutez la cellule pour effectuer l'analyse en générant le graphique à barres.



Jupyter AssessmentReviewNotebook Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted | conda_python3

questions aberrantes pour vérifier la qualité de la question elle-même ou déterminer si le matériel de formation original a besoin d'être amélioré.

```
In [17]: success_rate = dataset.groupby('questionId')['isCorrect'].mean()

plt.bar(success_rate.index, success_rate.values)

for i, rate in enumerate(success_rate):
    plt.text(i, rate, f'{rate:.0%}', ha='center', va='bottom')

plt.xlabel('Question ID')
plt.ylabel('Success Rate')
plt.title('Question Success Rate')

plt.xticks(rotation=90)

plt.show()
```

```
In [18]: success_rate = dataset.groupby('questionId')['isCorrect'].mean()

plt.bar(success_rate.index, success_rate.values)

for i, rate in enumerate(success_rate):
    plt.text(i, rate, f'{rate:.0%}', ha='center', va='bottom')

plt.xlabel('Question ID')
plt.ylabel('Success Rate')
plt.title('Question Success Rate')

plt.xticks(rotation=90)

plt.show()
```

