Spécification du modèle MCO avec statsmodels

```
import statsmodels.api as sm


# Spécification du modèle
X = df['hp']  # variable explicative
y = df['mpg']  # variable à prédire

# Ajout d'une constante à nos variables explicatives
X = sm.add_constant(X)

# Spécification du modèle MCO
model = sm.OLS(y, X)

# Estimation des paramètres
results = model.fit()

# Affichage des résultats
print(results.summary())
```

Hypotheses a valider

Pour que les résultats de l'analyse de régression linéaire soient valides, plusieurs hypothèses doivent être respectées:

1. **Linarité** : La relation entre la variable indépendante et la variable dépendante est linéaire.

```
import matplotlib.pyplot as plt
import statsmodels.api as sm

# Supposons que vous ayez ajusté un modèle OLS appelé "model"
model = sm.OLS(y, X).fit()

predictions = model.predict(X) # calcule des prédictions
residuals = model.resid # calcul des résidus

plt.scatter(predictions, residuals)
plt.axhline(0, color='red')
plt.xlabel('Valeurs prédites')
plt.ylabel('Résidus')
```

```
plt.title('Résidus vs Valeurs prédites')
plt.show()
```

Si les résidus sont distribués aléatoirement et le nuage de points est approximativement horizontal, cela indique que l'hypothèse de linéarité est respectée. Si les résidus montrent un motif ou une structure, cela pourrait indiquer que le modèle n'est pas approprié.

2. **Indépendance ou indépendance des erreurs** : Les observations sont indépendantes les unes des autres.
   Les valeurs du test de Durbin-Watson devraient être proches de 2 si les erreurs ne sont pas corrélées. `DurbinWatson` est disponible dans `statsmodels`.

```
from statsmodels.stats.stattools import durbin_watson

dw = durbin_watson(model.resid)
print('Durbin-Watson:', dw)
```

3. **Homoscédasticité** : La variance des erreurs est constante sur l'ensemble des niveaux de la variable indépendante. Cela signifie que la dispersion des résidus est la même quelles que soient les valeurs prises par la variable explicative.
   le test de Breusch-Pagan de `statsmodels` pour tester l'homoscédasticité:

```
from statsmodels.compat import lzip
from statsmodels.stats.diagnostic import het_breuschpagan

bp_test = het_breuschpagan(model.resid, model.model.exog)
labels = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']
lzip(labels, bp_test)
```

Le test de Breusch-Pagan retourne quatre valeurs :

i. **Lagrange multiplier statistic** : Il s'agit de la statistique du test de Breusch-Pagan.

ii. **p-value** : Cette valeur est utilisée pour accepter ou rejeter l'hypothèse nulle d'homoscédasticité. Si cette valeur p est inférieure au seuil de signification choisi (généralement 0,05), alors vous rejetez l'hypothèse nulle et concluez que vous avez des preuves suffisantes pour dire que les variances ne sont pas constantes (hétéroscédasticité). Si la valeur p est supérieure au seuil de signification, vous ne pouvez pas rejeter l'hypothèse nulle d'homoscédasticité.

iii. **f-value** : Il s'agit de la valeur F du test, qui est simplement une autre forme de la statistique du test.

iv. **f p-value** : Il s'agit de la valeur p associée à la valeur F du test. out comme la p-valeur du test de Lagrange, si cette p-valeur est inférieure au seuil de signification, alors nous rejetons l'hypothèse nulle et concluons qu'il y a une hétéroscédasticité significative dans le modèle.

4. **Normalité des erreurs** : Les erreurs (ou résidus) de la régression, c'est-à-dire la différence entre la valeur observée et la valeur prédite par le modèle pour chaque observation, suivent une distribution normale.
   une valeur faible pour la statistique de test de Jarque-Bera est une indication que les résidus ont une distribution proche de la normale (accompagnée d'une petite p-valeur, par exemple, moins de 0,05)
   L'hypothèse nulle pour ce test est que les résidus sont normalement distribués.

   ```
   ou:
   ```

```
sm.qqplot(model.resid, line ='45')
plt.show()
```

1. **Absence de multicollinéarité** : C'est à dire que les variables explicatives ne sont pas fortement corrélées entre elles. Cependant, dans une régression linéaire simple, cette condition est un peu moins pertinente car il n'y a qu'une seule variable explicative.

on peut vérifier l'absence de multicollinéarité en utilisant le facteur d'inflation de la variance (VIF):

```
from statsmodels.stats.outliers_influence import variance_inflation_factor

vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(X.values, i) for i in
range(X.shape[1])]
vif["features"] = X.columns
```

Notez que X est un DataFrame pandas qui contient vos variables explicatives. Les valeurs VIF supérieures à 5 peuvent indiquer un problème de multicollinéarité.

Differents test a faire

La régression des moindres carrés ordinaires (MCO) est une méthode statistique largement utilisée pour estimer les coefficients d'un modèle de régression. Pour tester la signification de ces coefficients, on utilise généralement les tests de Student et de Fisher.

1. **Test t** : Le test t est utilisé pour évaluer la signification de chaque coefficient de régression. Il teste l'hypothèse nulle que le coefficient est égal à zéro (c'est-à-dire que la variable n'a pas d'effet sur la réponse). Si la p-valeur du test t est inférieure à un seuil de signification (par exemple, 0.05), vous pouvez rejeter l'hypothèse nulle et conclure que la variable est significative.
2. **Test F** : Le test F est utilisé pour évaluer la signification globale du modèle de régression. Il teste l'hypothèse nulle que tous les coefficients de régression sont égaux à zéro (c'est-à-dire que aucune des variables n'a d'effet sur la réponse). Si la p-valeur du test F est inférieure à un seuil de signification, vous pouvez rejeter l'hypothèse nulle et conclure que au moins une des variables est significative.
3. **R-squared et Adjusted R-squared** : R-squared est une mesure de la proportion de la variance dans la variable dépendante qui est prédite par les variables indépendantes. Adjusted R-squared ajuste cette mesure pour le nombre de variables dans le modèle. Ces mesures peuvent vous aider à comprendre la "qualité d'ajustement" de votre modèle.
4. **AIC et BIC** : L'Akaike Information Criterion (AIC) et le Bayesian Information Criterion (BIC) sont des mesures de la qualité d'un modèle statistique. Elles tiennent compte à la fois de la qualité d'ajustement du modèle et du nombre de paramètres qu'il utilise. Lors de la comparaison de plusieurs modèles, un modèle avec un AIC ou un BIC plus bas est généralement préféré.
5. **Validation croisée** : Bien que ce ne soit pas un "test" à proprement parler, la validation croisée est une méthode importante pour évaluer la performance de votre modèle sur des données non vues. Elle peut vous aider à comprendre comment votre modèle se généralisera à de nouvelles données.
   on utilise scikit-learn
6. **Intervalles de confiance** : Les intervalles de confiance pour les coefficients de régression donnent une gamme de valeurs dans laquelle vous pouvez être "confiant" que les vrais coefficients se trouvent.

```
print(results.conf_int())
```

La méthode `.summary()` donne une sortie qui comprend :

- Les coefficients de régression pour chaque variable dans le modèle (colonne `coef`)
- Les valeurs t et les p-valeurs pour le test t pour chaque coefficient (colonnes `t` et `P>|t|`)
- La statistique F et la p-valeur pour le test F du modèle dans son ensemble (`F-statistic` et `Prob (F-statistic)`)

- Le R-squared et Adjusted R-squared du modèle (`R-squared` et `Adj. R-squared`)
- La statistique de Jarque-Bera et la p-valeur pour le test de normalité des résidus (`Jarque-Bera (JB)` et `Prob(JB)`)
- L'AIC et le BIC du modèle (`AIC` et `BIC`)

details:

1. **Test de Student (t-test)**

Le test de Student permet de tester l'hypothèse nulle que le coefficient de la variable indépendante dans la régression est égal à zéro (c'est-à-dire que la variable n'a pas d'effet sur la variable dépendante). La statistique de test est donnée par :

$t = \hat{\beta} / SE(\hat{\beta})$

où $\hat{\beta}$ est l'estimateur des moindres carrés du coefficient de régression, et $SE(\hat{\beta})$ est l'erreur standard de $\hat{\beta}$. Sous l'hypothèse nulle, cette statistique suit une distribution t de Student avec (n - k) degrés de liberté, où n est le nombre d'observations et k est le nombre de variables indépendantes (plus une pour l'intercept).

2. **Test de Fisher (F-test)**

Le test de Fisher est utilisé pour tester l'hypothèse nulle que tous les coefficients de régression sont égaux à zéro (c'est-à-dire que aucune des variables indépendantes n'a d'effet sur la variable dépendante). La statistique de test est donnée par :

$F = [(SSR / k) / (SSE / (n - k - 1))]$

où SSR est la somme des carrés de la régression (la variation expliquée par le modèle), SSE est la somme des carrés des erreurs (la variation non expliquée par le modèle), n est le nombre d'observations, et k est le nombre de variables indépendantes. Sous l'hypothèse nulle, cette statistique suit une distribution F avec k et (n - k - 1) degrés de liberté.

Test de robustesse

Quelques tests de sensibilité et de robustesse comprennent :

1. **Analyse de sensibilité** : Cela consiste à modifier les valeurs des variables d'entrée et à observer l'effet sur la variable de sortie. Par exemple, si vous disposez d'un modèle de régression linéaire, vous pouvez modifier les valeurs des variables indépendantes une par une et observer l'effet sur la variable dépendante.
2. **Validation croisée** : La validation croisée est une technique pour évaluer la robustesse d'un modèle en testant sa performance sur différentes sous-parties des données. Les

méthodes courantes de validation croisée comprennent la validation croisée k-fold et la validation croisée Leave-One-Out (LOO).

3. **Bootstrap** : Le bootstrap est une technique de rééchantillonnage qui peut être utilisée pour estimer la variabilité et la robustesse d'un estimateur statistique. Il s'agit de créer de nombreux échantillons à partir des données d'origine, chaque échantillon étant sélectionné avec remplacement, puis d'ajuster le modèle à chaque échantillon et d'observer la variabilité dans les estimations.

```
from statsmodels.graphics.gofplots import qqplot
from statsmodels.graphics.tsaplots import plot_acf
import numpy as np
import matplotlib.pyplot as plt
import statsmodels.api as sm

# Vos données
X = sm.add_constant(X)
model = sm.OLS(y, X)
results = model.fit()

# Bootstrap des résidus
n_bootstrap = 1000
bootstrap_results = []
for _ in range(n_bootstrap):
    sample_residuals = np.random.choice(results.resid, size=len(y), replace=True)
    sample_y = results.predict(X) + sample_residuals
    bootstrap_model = sm.OLS(sample_y, X)
    bootstrap_results.append(bootstrap_model.fit().params)
```

4. **Analyse des résidus** : Analyser les résidus d'un modèle (la différence entre les valeurs prédites et les valeurs réelles) peut fournir des informations sur la robustesse du modèle. Par exemple, si les résidus sont bien distribués autour de zéro et ne montrent pas de motifs évidents, cela peut indiquer que le modèle est robuste.

```
residuals = results.resid
plt.scatter(results.fittedvalues, residuals)
plt.axhline(y=0, color='r', linestyle='-')
```

5. **Analyse d'influence** : L'analyse d'influence est une technique pour évaluer l'effet de chaque observation sur les estimations du modèle. Cela peut aider à identifier les points de données qui ont une influence disproportionnée sur le modèle.

```
from statsmodels.stats.outliers_influence import OLSInfluence

influence = OLSInfluence(results)
fig, ax = plt.subplots()
ax.scatter(influence.hat_matrix_diag, influence.resid_studentized_internal)
```

6. **Vérification des hypothèses du modèle** : Pour un modèle de régression linéaire, vous voudriez vérifier les hypothèses de linéarité, d'indépendance des erreurs, d'homoscédasticité, et de normalité des erreurs. Si ces hypothèses sont violées, cela peut indiquer que votre modèle n'est pas robuste.
7. **Tests de spécification du modèle** : Ces tests examinent si le modèle a été correctement spécifié. Par exemple, ils peuvent vérifier si les variables importantes ont été omises, si les variables inutiles ont été incluses, ou si la forme fonctionnelle du modèle est correcte.
8. **Test de la stabilité temporelle** : Si vous travaillez avec des données de série temporelle, vous pouvez diviser vos données en différentes périodes de temps et vérifier si les coefficients de votre modèle restent stables au fil du temps.

Previsions et analyses

utiliser la méthode `predict()` de l'objet `results`:

```
import numpy as np
import statsmodels.api as sm

# Ajuster le modèle
X = sm.add_constant(X)  # Ajoute une constante (intercept) à vos données
model = sm.OLS(y, X)
results = model.fit()

# Prédire de nouvelles valeurs
new_X_values = np.array([1, valeur_de_variable_exogene])  # Ajoute 1 pour
l'intercept
prediction = results.predict(new_X_values)

print("La valeur prédite de la variable endogène pour x =",
valeur_de_variable_exogene, "est", prediction)
```

pour plusieurs valeurs:

```
new_X_values = sm.add_constant(new_X_values)  # Ajoute une colonne de 1 pour
l'intercept
```

```
predictions = results.predict(new_X_values)  # Faire des prédictions pour toutes les nouvelles observations
```

ou

```
import numpy as np
import statsmodels.api as sm

# Générons quelques données pour cet exemple
np.random.seed(0)
X = np.random.rand(100, 1)
y = 3*X.squeeze() + 2 + np.random.randn(100)  # y = 3x + 2 + bruit

# Ajuster le modèle
X = sm.add_constant(X)  # ajoute une constante (intercept) à vos données
model = sm.OLS(y, X)
results = model.fit()

# Maintenant, disons que nous avons plusieurs nouvelles valeurs de x (la variable exogène) pour lesquelles nous voulons prédire y
new_x_values = np.array([[0.5], [0.6], [0.7]])
new_X_values = sm.add_constant(new_x_values)  # Ajoute une colonne de 1 pour l'intercept

# Faire la prédiction
predictions = results.predict(new_X_values)

for i, new_x in enumerate(new_x_values[:, 1]):  # le deuxième élément de chaque ligne est la valeur de x
    print("La valeur prédite de la variable endogène pour x =", new_x, "est", predictions[i])
```

analyses

```
from sklearn.metrics import mean_squared_error
from math import sqrt

real_values = ...  # Remplacez par vos valeurs réelles
rmse = sqrt(mean_squared_error(real_values, predictions))
```

ANNEXES

t-Student

La régression linéaire simple est définie comme une régression linéaire avec une seule variable prédictive. Un exemple de régression linéaire simple est Y = aX + b.

La linéarité de la relation linéaire peut être déterminée en calculant la statistique du test t . La statistique t aide à déterminer dans quelle mesure cette relation linéaire est linéaire ou non linéaire. Examinons la formulation de l'hypothèse en relation avec la détermination de la relation entre les variables dépendantes et indépendantes et la façon dont la valeur que les coefficients prennent pour quantifier la relation.

1. **Hypothèse nulle (H_0)** : L'hypothèse nulle indique qu'il n'y a pas de relation entre la caractéristique (variable indépendante/prédictive) et la variable dépendante/réponse. En termes de coefficients, cela suggère que le **coefficient de la caractéristique est égal à zéro** .
2. **Hypothèse alternative (H_1)** : L'hypothèse alternative contredit l'hypothèse nulle et suggère qu'il existe une relation entre la caractéristique et la variable dépendante. Cela implique que le **coefficient de la caractéristique n'est pas égal à zéro** .

```
- H0 : a = 0
- Ha : a ≠ 0
```

La formule de calcul de la statistique t dans le contexte de la régression linéaire est la suivante:

```
t = (β – 0) / SE(β)
```

- t est la statistique t
- β est le coefficient estimé de la caractéristique. S'il y a plusieurs caractéristiques, nous aurons des valeurs telles que β1, β2, β3, …, βn. De cette façon, nous aurons des statistiques t pour chacun des coefficients.
- 0 est la valeur hypothétique (généralement zéro sous l'hypothèse nulle)
- SE(β) est l'erreur type du coefficient estimé

calculer la valeur p et comparer avec le niveau de signification (0,05). Si la valeur de p inférieure à 0,05 indique que le résultat du test t est statistiquement significatif et que vous pouvez rejeter l'hypothèse nulle selon laquelle il n'y a pas de relation entre Y et X (a=0). Cela signifie que la valeur de a calculée est valable.