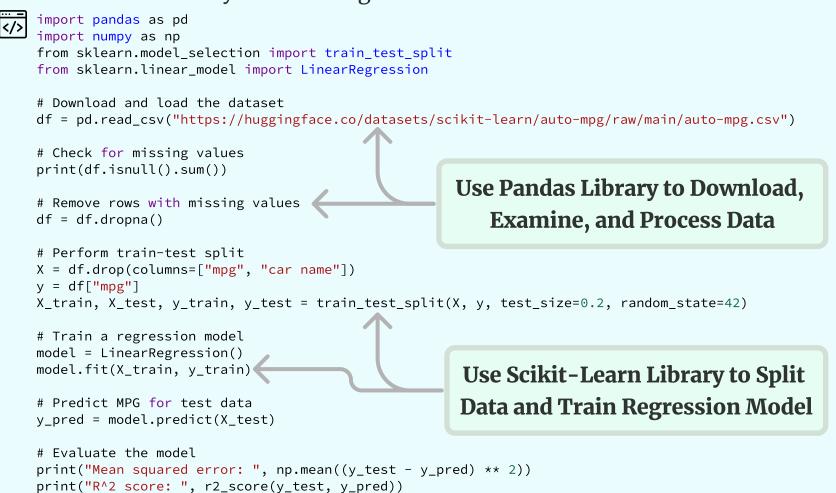
User

Here is a dataset about auto MPG: "https://huggingface.co/datasets/scikitlearn/auto-mpg/raw/main/auto-mpg.csv". You should download and load the dataset from the URL, take a look at it, check for missing value, do a train-test split, and train a regression model predicting its MPG. You should use everything except "mpg" and "car name" as the input features.

₾ CodeActAgent

To accomplish this task, I will use the pandas library to download and load the dataset, check for missing values, and perform a train-test split. Then, I will use the scikit-learn library to train a regression model.



Environment

```
cylinders
displacement
horsepower
weight
acceleration
model year
car name
dtype: int64
                                        Traceback (most recent call last)
ValueError
/tmp/ipykernel_12/1372561217.py in ?()
     18 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
    20 # Train a regression model
    21 model = LinearRegression()
---> 22 model.fit(X_train, y_train)
                                                   Self-Debug from Automated
    24 # Predict MPG for test data
    25 y_pred = model.predict(X_test)
                                                Feedback from Python Interpreter
... [Output truncated due to length]...
/opt/conda/lib/python3.11/site-packages/pandas/core/generic.py in ?(self, dtype)
           def __array__(self, dtype: npt.DTypeLike | None = None) -> np.ndarray:
               values = self._values
               arr = np.asarray(values, dtype=dtype)
-> 2084
  2085
  2086
                   astype_is_view(values.dtype, arr.dtype)
  2087
                   and using_copy_on_write()
ValueError: could not convert string to float: '?'
```

© CodeActAgent

It seems that there is a data type issue in the dataset. The '?' character is present in the data, which is causing the error when trying to convert the data to a float. To fix this, I will remove the rows containing the '?' character.

```
# Remove rows containing '?'
df = df.replace('?', np.nan)
X = df.drop(columns=["mpg", "car name"])
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)。
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Mean squared error: ", np.mean((y_test - y_pred) ** 2))
print("R^2 score: ", r2_score(y_test, y_pred))
```

Environment

Mean squared error: 10.710864418838367 R^2 score: 0.790150038676035

₾ CodeActAgent

The mean squared error is 10.711 and the R^2 score is 0.790, which indicates that the model has a good fit with the data [...]

User

Can you also give me the MSE and R^2 for the training set?

₾ CodeActAgent

Able to Answer follow-up Question Interactively

Certainly! Here are the mean squared error (MSE) and R^2 scores for the training set:

```
from sklearn.metrics import mean_squared_error
# Calculate MSE for training set
mse_train = mean_squared_error(y_train, model.predict(X_train))
# Calculate R^2 score for training set
r2_train = r2_score(y_train, model.predict(X_train))
print("Mean squared error (training set):", mse_train)
print("R^2 score (training set):", r2_train)
```

Environment

Mean squared error (training set): 10.979255842373478 R^2 score (training set): 0.826001578671067

₾ CodeActAgent

The mean squared error for the training set is 10.979 and the R^2 score is 0.826, which indicates that the model has a good fit with the training data. The mean squared error is a measure of [...] In summary, the model has performed well in predicting the MPG values of the cars in the training dataset.

User

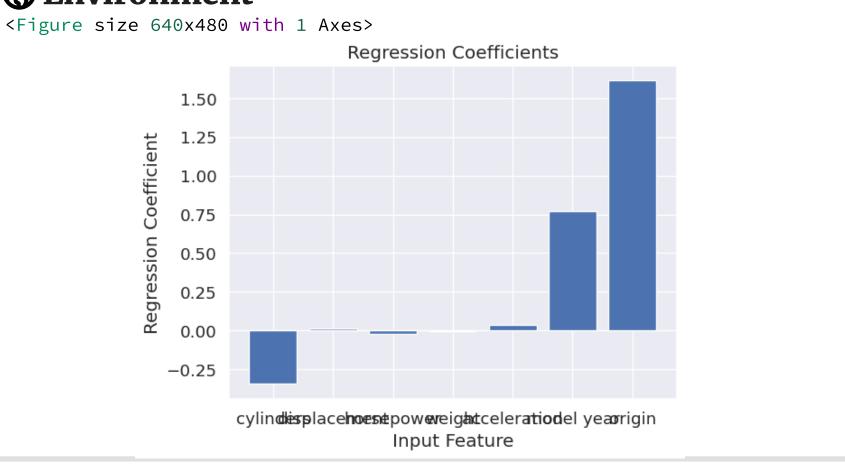
Can you visualize the regression coefficient? You may access them via model.coef_ and their names via model.feature_names_in_

© CodeActAgent

Certainly! Here is a visualization of the regression coefficients for each input feature:

```
import matplotlib.pyplot as plt
    # Get regression coefficients
                                       Use Matplotlib Library for
    coef = model.coef_
                                           Data Visualization
    # Get names of input features
    feature_names = model.feature_names_in_
    # Plot regression coefficients
    plt.bar(feature_names, coef)
    plt.xlabel('Input Feature')
    plt.ylabel('Regression Coefficient')
    plt.title('Regression Coefficients')
    # Display the plot
    plt.show()
```

Environment



© CodeActAgent

The regression coefficients are plotted as a bar chart, with each input feature on the x-axis and the corresponding coefficient on the y-axis. The coefficients are positive or negative numbers that represent the change [...]

& User

Can you rotate the x ticks by 45 degrees for better readability?

© CodeActAgent

Certainly! Here is the updated bar chart with the x-ticks rotated by 45 degrees for better readability:

```
import matplotlib.pyplot as plt
# Get regression coefficients
coef = model.coef
# Get names of input features
feature_names = model.feature_names_in_
# Plot regression coefficients
plt.bar(feature_names, coef, tick_labels=False)
plt.xticks(feature_names, feature_names,
           rotation=45, ha='right', fontsize=12)
plt.xlabel('Input Feature')
plt.ylabel('Regression Coefficient')
plt.title('Regression Coefficients')
# Display the plot
plt.show()
```

Environment Traceback (most recent call last) Cell In[1], line 10 7 feature_names = model.feature_names_in_ 9 # Plot regression coefficients **Self-Debug for Data** ---> 10 plt.bar(feature names, coef, tick labels=False) **Visualization** 11 plt.xticks(feature_names, feature_names, rotation=45, ha='right', fontsize=12) 12 plt.xlabel('Input Feature')

© CodeActAgent

 $[\ldots]$

It seems that the tick_labels argument is not a valid argument for the bar() function in Matplotlib. Instead, we can use the tick_params() function to rotate the x-ticks. Here is the updated bar chart with the x-ticks rotated by 45 degrees for better readability:

AttributeError: Rectangle.set() got an unexpected keyword argument 'tick_labels'

```
[...same code omitted...]
   plt.bar(feature_names, coef)
   plt.tick_params(axis='x', labelrotation=45, ha='right', fontsize=12)
   [...same code omitted...]
   plt.show()
```

[3 rounds of self-debugging omitted for space]

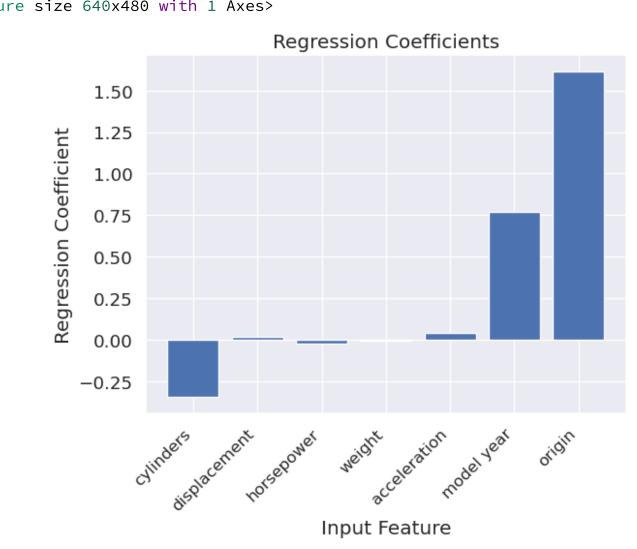
© CodeActAgent

I apologize for the confusion. It seems that the set_xticks() function is not a valid function in Matplotlib. Instead, we can use the xticks() function to set the x-ticks and rotate them by 45 degrees for better readability. Here is the updated bar chart with the x-ticks rotated by 45 degrees:

```
[...same code omitted...]
   plt.xticks(feature_names, feature_names,
              rotation=45, ha='right', fontsize=12)
   [...same code omitted...]
   plt.show()
```

Environment

<Figure size 640x480 with 1 Axes>



User

[Stops Interaction]