

Gemini 2.0 Flash Thinking

DeepSeek-R1

```
import ...
```

df = pd.read_csv('unemployement_industry.csv')

X = df['Mar.2019'].values.reshape(-1, 1)

y = df['Mar.2020'].values.reshape(-1, 1)

imputer = SimpleImputer(strategy='mean')

Logical error injected here

X = imputer.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(X, y,

test_size=0.3, random_state=42)

model = LinearRegression()

model.fit(X_train, y_train)

y pred = model.predict(X test)

mse = mean_squared_error(y_test, y_pred)

mse rounded = round(mse, 2)

The error is likely in the data preprocessing or model training part.... X=imputer.fit_transform(y). This line is suspicious.

. . .

However, the prompt asks for the interpreter error. ... This seems contradictory. If it's a logic error, it won't throw an interpreter error.

•••

If y contains non-numeric values, then imputer.fit(y) might fail. ...But the data is likely numeric. Let's assume data is numeric... If y contains strings that cannot be converted to float, then imputer.fit(y) will fail.

•••

Assuming 'Mar.2020' column has a string like 'N/A', then imputer.fit(y) will fail with ValueError: could not convert string to float.

The line says X = imputer.fit_transform(y). Wait, that's odd.

•••

However, the problem states that there's an error that the Python interpreter would throw. So perhaps there's another error...

•••

Ah! Wait, the code uses X and y after imputing. But after imputing, X is based on y (Mar.2020), and y is the original Mar.2020... If the original y (Mar.2020) has missing values, then y would contain NaNs... When the model is trained on y_train (which has NaNs), the LinearRegression's fit method would throw an error because it cannot handle NaN values in y. In this case, the error would occur at the line model.fit(X_train, y_train) with a ValueError: Input contains NaN. ✓



Both Models:

- Quickly Identifying the Cause Error Line
- Systematic Code Execution Tracing
- Exploration of Potential Error Scenarios
- Iterative Refinement of Hypotheses

Gemini Flash Thinking:

• Sidetracked by an early theory

DeepSeek-R1:

- More Persistent Pursuit of Runtime Error.
- "Aha!" Moment leading to the Intended Error.
- Demonstrated stronger reasoning ability to find the subtle runtime error.

Overall:

- Both models are highly capable and effectively used simulated code execution tracing and scenario exploration.
- The "interpreter error" constraint was a crucial driver for both models' reasoning, pushing them beyond simply identifying the cause error line and towards finding a runtime manifestation.