```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

%matplotlib inline
```

```
!unzip archive.zip
```

```
Archive:  archive.zip
  inflating: ai4i2020.csv
```

```
df = pd.read_csv("ai4i2020.csv")
df.head()
```

	UDI	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Machine failure	TWF	HDF	PWF	OSF	RNF
0	1	M14860	M	298.1	308.6	1551	42.8	0	0	0	0	0	0	0
1	2	L47181	L	298.2	308.7	1408	46.3	3	0	0	0	0	0	0
2	3	L47182	L	298.1	308.5	1498	49.4	5	0	0	0	0	0	0
3	4	L47183	L	298.2	308.6	1433	39.5	7	0	0	0	0	0	0
4	5	L47184	L	298.2	308.7	1408	40.0	9	0	0	0	0	0	0

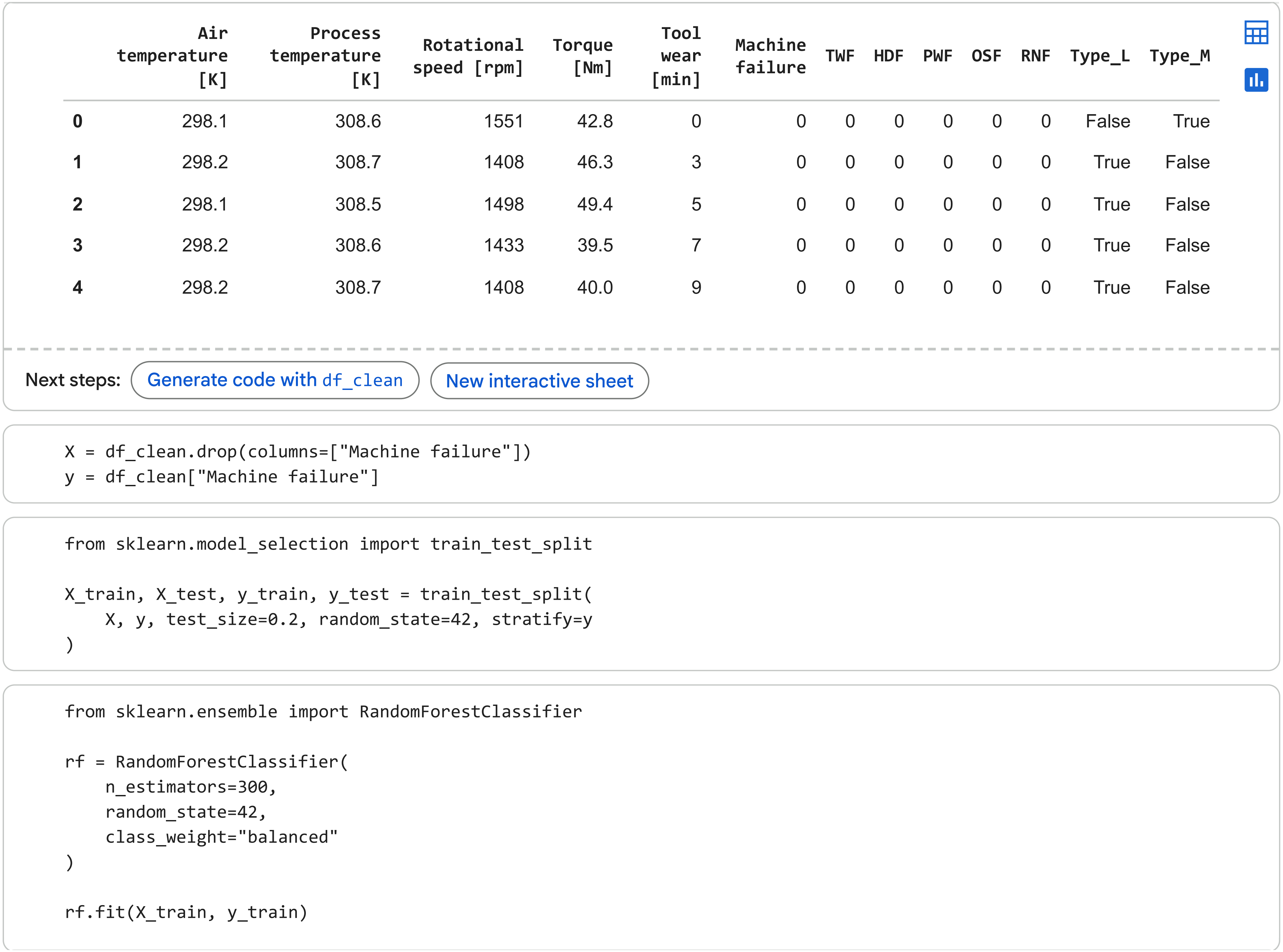
Next steps: Generate code with df New interactive sheet

```
# Drop ID columns that don't help prediction
df_clean = df.drop(columns=["UDI", "Product ID"])

df_clean.head()
```

	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Machine failure	TWF	HDF	PWF	OSF	RNF
0	M	298.1	308.6	1551	42.8	0	0	0	0	0	0	0
1	L	298.2	308.7	1408	46.3	3	0	0	0	0	0	0
2	L	298.1	308.5	1498	49.4	5	0	0	0	0	0	0
3	L	298.2	308.6	1433	39.5	7	0	0	0	0	0	0
4	L	298.2	308.7	1408	40.0	9	0	0	0	0	0	0

Next steps: Generate code with `df_clean` New interactive sheet

```
# Convert Type (L,M,H) into numeric dummy variables
df_clean = pd.get_dummies(df_clean, columns=["Type"], drop_first=True)
df_clean.head()
```



	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Machine failure	TWF	HDF	PWF	OSF	RNF	Type_L	Type_M
0	298.1	308.6	1551	42.8	0	0	0	0	0	0	0	False	True
1	298.2	308.7	1408	46.3	3	0	0	0	0	0	0	True	False
2	298.1	308.5	1498	49.4	5	0	0	0	0	0	0	True	False
3	298.2	308.6	1433	39.5	7	0	0	0	0	0	0	True	False
4	298.2	308.7	1408	40.0	9	0	0	0	0	0	0	True	False

```
X = df_clean.drop(columns=["Machine failure"])
y = df_clean["Machine failure"]
```

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(
    n_estimators=300,
    random_state=42,
    class_weight="balanced"
)

rf.fit(X_train, y_train)
```

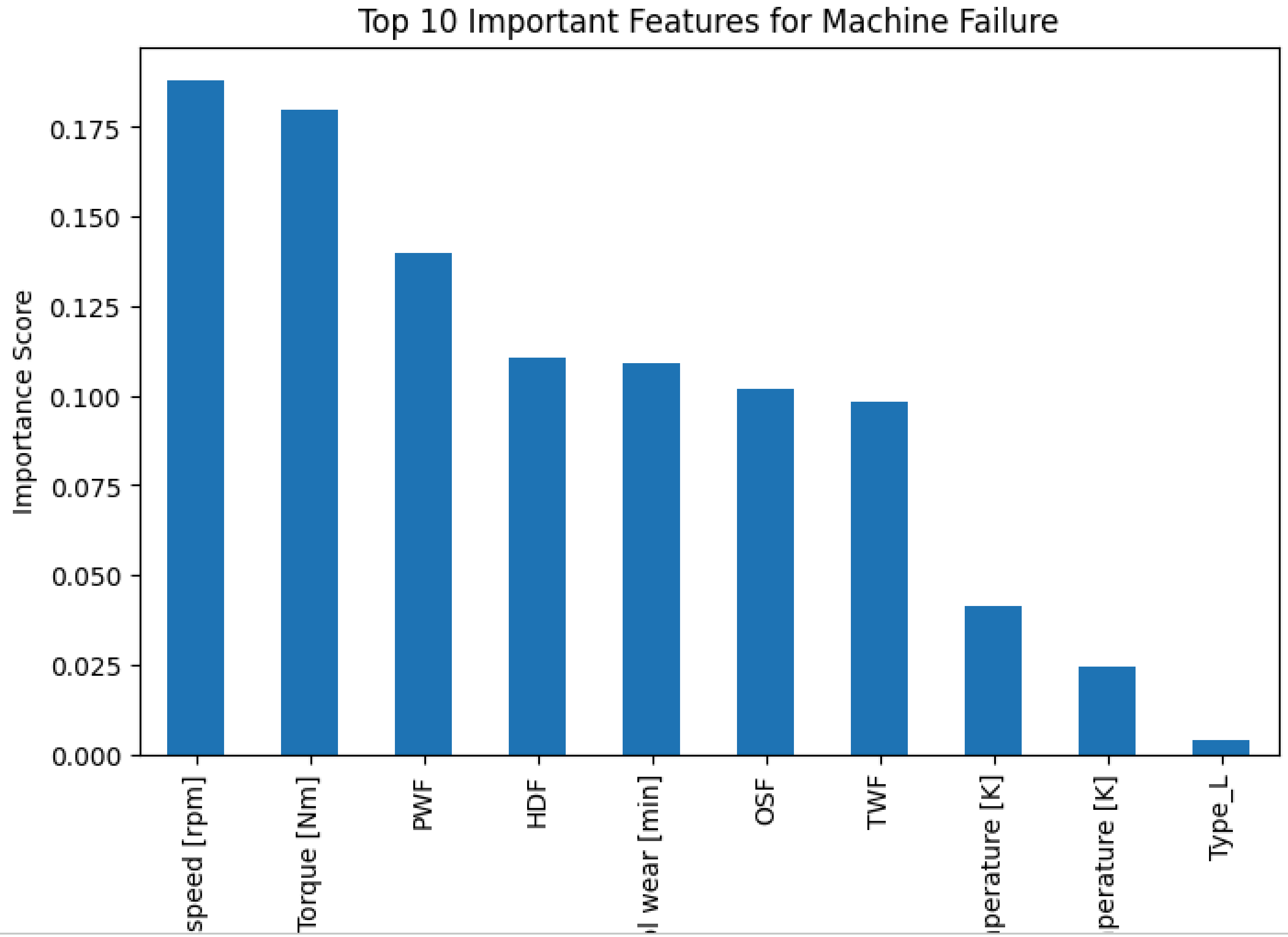
▾ RandomForestClassifier ⓘ ?

```
RandomForestClassifier(class_weight='balanced', n_estimators=300,
                       random_state=42)
```

```python
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

y_pred = rf.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
Accuracy: 0.9985

Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00      1932
           1       1.00      0.96      0.98        68

    accuracy                           1.00      2000
   macro avg       1.00      0.98      0.99      2000
weighted avg       1.00      1.00      1.00      2000


Confusion Matrix:
 [[1932    0]
 [   3   65]]
```

```python
importances = pd.Series(rf.feature_importances_, index=X.columns)
top10 = importances.sort_values(ascending=False).head(10)

plt.figure(figsize=(8,5))
top10.plot(kind="bar")
plt.title("Top 10 Important Features for Machine Failure")
plt.ylabel("Importance Score")
plt.show()
plt.show()
```



Double-click (or enter) to edit