```
from google.colab import files
uploaded = files.upload()
```

Choose Files Pearl Chall…hare_v4.xlsx

- **Pearl Challenge data with dictionary_For_Share_v4.xlsx**(application/vnd.openxmlformats-officedocument.spreadsheetml.sheet) - 46097184 bytes, last modified: 5/12/2025 - 100% done

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.impute import SimpleImputer

# Load Excel file and check sheets
xls = pd.ExcelFile('Pearl Challenge data with dictionary_For_Share_v4.xlsx')
print(xls.sheet_names)  # Optional: verify sheet names

# Load the correct sheet
df = pd.read_excel(xls, sheet_name='TrainData')  # ✅ Use correct sheet

# Drop unnecessary columns (like IDs if present)
df = df.drop(columns=['FarmerID'], errors='ignore')  # Modify if needed

# Set target column
target_column = 'Target_Variable/Total Income'
X = df.drop(columns=[target_column])
y = df[target_column]

# Handle missing values
cat_cols = X.select_dtypes(include=['object']).columns
num_cols = X.select_dtypes(include=['number']).columns

cat_imputer = SimpleImputer(strategy='most_frequent')
num_imputer = SimpleImputer(strategy='mean')

X[cat_cols] = cat_imputer.fit_transform(X[cat_cols])
X[num_cols] = num_imputer.fit_transform(X[num_cols])

# Label encode categorical variables
encoder = LabelEncoder()
for col in cat_cols:
    X[col] = encoder.fit_transform(X[col])

# Train/test split
X_train, X_val, y_train, y_val = train_test_split(X, y, test_size=0.2, random_state=42)

# Train model
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Evaluate
y_pred = model.predict(X_val)
rmse = np.sqrt(mean_squared_error(y_val, y_pred))
print(f"Validation RMSE: {rmse:.2f}")
```

```
['TrainData', 'TestData', 'Dictionary']
Validation RMSE: 627944.33
```

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestRegressor
```

```
# Load Excel file and check sheets
xls = pd.ExcelFile('Pearl Challenge data with dictionary_For_Share_v4.xlsx')
print(xls.sheet_names)  # Optional: verify sheet names

# Load the correct sheet
df = pd.read_excel(xls, sheet_name='TrainData')  # ✅ Use correct sheet

# Drop unnecessary columns (like IDs if present)
df = df.drop(columns=['FarmerID'], errors='ignore')  # Modify if needed

# Set target column
target_column = 'Target_Variable/Total Income'
X = df.drop(columns=[target_column])
y = df[target_column]

# Handle missing values
cat_cols = X.select_dtypes(include=['object']).columns
num_cols = X.select_dtypes(include=['number']).columns

cat_imputer = SimpleImputer(strategy='most_frequent')
num_imputer = SimpleImputer(strategy='mean')

X[cat_cols] = cat_imputer.fit_transform(X[cat_cols])
X[num_cols] = num_imputer.fit_transform(X[num_cols])

# Label encode categorical variables
encoder = LabelEncoder()
for col in cat_cols:
    X[col] = encoder.fit_transform(X[col])

# Train model with optimizations: reduce n_estimators, add max_depth, use parallel processing
model = RandomForestRegressor(n_estimators=10, random_state=42, max_depth=10, n_jobs=-1)
model.fit(X, y)

# Load test data and drop unnecessary columns
test_df = pd.read_excel(xls, sheet_name='TestData')

# Ensure the test data has the same columns as the training data (order and names)
test_df = test_df[X.columns]  # Align the columns in test data to the training data

# Handle missing values in the test data using the same imputers from training data
test_cat_cols = test_df.select_dtypes(include=['object']).columns
test_num_cols = test_df.select_dtypes(include=['number']).columns

# Impute missing categorical values
test_df[test_cat_cols] = cat_imputer.transform(test_df[test_cat_cols])

# Impute missing numerical values
test_df[test_num_cols] = num_imputer.transform(test_df[test_num_cols])

# Label encode categorical columns in the test set using the same encoder
for col in test_cat_cols:
    test_df[col] = encoder.transform(test_df[col])

# Predict using the trained model
test_predictions = model.predict(test_df)

# Step 5: Save the predictions to a new DataFrame with FarmerID
test_ids = pd.read_excel(xls, sheet_name='TestData')['FarmerID']
output_df = pd.DataFrame({
    'FarmerID': test_ids,
    'Predicted_Income': test_predictions
})

# Step 6: Save the output predictions to a CSV file
output_df.to_csv('test_predictions.csv', index=False)
print("✅ Predictions saved to test_predictions.csv")
```

```
import pandas as pd
import numpy as np
from sklearn.impute import SimpleImputer
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import OrdinalEncoder

# Load Excel file
xls = pd.ExcelFile('Pearl Challenge data with dictionary_For_Share_v4.xlsx')

# Load training data
train_df = pd.read_excel(xls, sheet_name='TrainData')
train_df = train_df.drop(columns=['FarmerID'], errors='ignore')

# Set target and features
target_column = 'Target_Variable/Total Income'
X_train = train_df.drop(columns=[target_column])
y_train = train_df[target_column]

# Separate categorical and numerical columns
cat_cols = X_train.select_dtypes(include=['object']).columns
num_cols = X_train.select_dtypes(include=['number']).columns

# Imputers
cat_imputer = SimpleImputer(strategy='most_frequent')
num_imputer = SimpleImputer(strategy='mean')

X_train[cat_cols] = cat_imputer.fit_transform(X_train[cat_cols])
X_train[num_cols] = num_imputer.fit_transform(X_train[num_cols])

# Use OrdinalEncoder and handle unknowns
encoder = OrdinalEncoder(handle_unknown='use_encoded_value', unknown_value=-1)
X_train[cat_cols] = encoder.fit_transform(X_train[cat_cols])

# Train model
model = RandomForestRegressor(n_estimators=10, random_state=42, max_depth=10, n_jobs=-1)
model.fit(X_train, y_train)

# Load test data
test_df = pd.read_excel(xls, sheet_name='TestData')
test_ids = test_df['FarmerID']  # Save IDs before dropping

# Ensure same columns
test_df = test_df[X_train.columns]

# Impute missing values
test_df[cat_cols] = cat_imputer.transform(test_df[cat_cols])
test_df[num_cols] = num_imputer.transform(test_df[num_cols])

# Encode test categorical features
test_df[cat_cols] = encoder.transform(test_df[cat_cols])

# Predict
predictions = model.predict(test_df)

# Output to CSV
output_df = pd.DataFrame({
    'FarmerID': test_ids,
    'Predicted_Income': predictions
})
output_df.to_csv('salary_predictions.csv', index=False)
print("✅ Predictions saved to salary_predictions.csv")
```

```
✅ Predictions saved to salary_predictions.csv
```

```
# To preview in notebook or script
import pandas as pd
predictions_df = pd.read_csv('salary_predictions.csv')
```

```
print(predictions_df.head())
```

```
            FarmerID  Predicted_Income
0   576972022499073      1.242662e+06
1   979235081831136      7.754480e+05
2   176490610549774      7.606357e+05
3   977021407171384      1.382900e+06
4  1334154133262320      1.345066e+06
```

```
print(predictions_df['Predicted_Income'].describe())
```

```
count    1.000000e+04
mean     1.231514e+06
std      2.233813e+06
min      7.348535e+05
25%      8.838522e+05
50%      1.021344e+06
75%      1.208830e+06
max      1.126630e+08
Name: Predicted_Income, dtype: float64
```

```
# Check how many predictions were made
print("Total farmers in test set:", len(predictions_df))
```

```
Total farmers in test set: 10000
```

```
import pandas as pd

xls = pd.ExcelFile('Pearl Challenge data with dictionary_For_Share_v4.xlsx')
test_ids = pd.read_excel(xls, sheet_name='TestData')['FarmerID']

print("Original test farmers:", len(test_ids))
print("Predicted farmers:", len(predictions_df))
```

```
Original test farmers: 10000
Predicted farmers: 10000
```

```
import pandas as pd

df = pd.read_csv('salary_predictions.csv')
print(df.head())  # Show first few rows
```

```
            FarmerID  Predicted_Income
0   576972022499073      1.242662e+06
1   979235081831136      7.754480e+05
2   176490610549774      7.606357e+05
3   977021407171384      1.382900e+06
4  1334154133262320      1.345066e+06
```

```
from google.colab import files
files.download('salary_predictions.csv')
```