
Lab 5: Data Quality and Validation

**Date: Apr 11, 2025**

Ryan Varghese, 100870665

# 1. Great Expectations

```
[ ] df.head()
```

| | age | workclass | fnlwgt | education | education-num | marital-status | occupation | relationship | race | sex | capital-gain | capital-loss | hours-per-week | native-country | income |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 39 | State-gov | 77516 | Bachelors | 13 | Never-married | Adm-clerical | Not-in-family | White | Male | 2174 | 0 | 40 | United-States | <=50K |
| 1 | 50 | Self-emp-not-inc | 83311 | Bachelors | 13 | Married-civ-spouse | Exec-managerial | Husband | White | Male | 0 | 0 | 13 | United-States | <=50K |
| 2 | 38 | Private | 215646 | HS-grad | 9 | Divorced | Handlers-cleaners | Not-in-family | White | Male | 0 | 0 | 40 | United-States | <=50K |
| 3 | 53 | Private | 234721 | 11th | 7 | Married-civ-spouse | Handlers-cleaners | Husband | Black | Male | 0 | 0 | 40 | United-States | <=50K |
| 4 | 28 | Private | 338409 | Bachelors | 13 | Married-civ-spouse | Prof-specialty | Wife | Black | Female | 0 | 0 | 40 | Cuba | <=50K |

```
validation_result = batch.validate(expectation)
print(validation_result)
```

```
Calculating Metrics: 100% 10/10 [00:00<00:00, 220.28it/s]
```


```
{
  "success": true,
  "expectation_config": {
    "type": "expect_column_values_to_be_between",
    "kwargs": {
      "batch_id": "pandas-pd dataframe asset",
      "column": "education-num",
      "min_value": 0.0,
      "max_value": 20.0
    },
    "meta": {}
  },
  "result": {
    "element_count": 32561,
    "unexpected_count": 0,
    "unexpected_percent": 0.0,
    "partial_unexpected_list": [],
    "missing_count": 0,
    "missing_percent": 0.0,
    "unexpected_percent_total": 0.0,
    "unexpected_percent_nonmissing": 0.0,
    "partial_unexpected_counts": [],
    "partial_unexpected_index_list": []
  },
  "meta": {},
  "exception_info": {
    "raised_exception": false,
    "exception_traceback": null,
    "exception_message": null
  }
}
```


Task 1:

```
[4] df.head()
```

| | Timestamp | Car1_location_X | Car1_location_Y | Car1_location_Z | Car2_location_X | Car2_location_Y | Car2_location_Z | Occluded_Image_view | Occluding_Car_view | Ground_Truth_View | pede |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1736796157 | -51.402977 | 143 | 0.596902 | -59.320270 | 140 | 0.596902 | A_001.png | B_001.png | C_001.png | |
| 1 | 1736796167 | -53.819637 | 143 | 0.596902 | -59.196568 | 140 | 0.596902 | A_002.png | B_002.png | C_002.png | |
| 2 | 1736796178 | -50.239144 | 143 | 0.596902 | -56.744479 | 140 | 0.596902 | A_003.png | B_003.png | C_003.png | |
| 3 | 1736796188 | -53.707220 | 143 | 0.596902 | -57.309380 | 140 | 0.596902 | A_004.png | B_004.png | C_004.png | |
| 4 | 1736796198 | -52.053721 | 143 | 0.596902 | -59.545897 | 140 | 0.596902 | A_005.png | B_005.png | C_005.png | |

```
context = gx.get_context()
datasource = context.data_sources.add_pandas(name="my_pandas_datasource")
```

```
INFO:great_expectations.data_context.types.base:Created temporary directory '/tmp/tmphuhl0qwj' for ephemeral docs
```

## 2. CleanLab

Task 2:

```
# Define a Random Forest classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# Use Cleanlab's CleanLearning
cleaner = CleanLearning(clf)
cleaner.fit(X_train, y_train)
```

```
CleanLearning(clf=RandomForestClassifier(random_state=42),
              find_label_issues_kwargs={'confident_joint': array([[40,  0,  0],
       [ 0, 40,  1],
       [ 0,  0, 39]]),
                                        'min_examples_per_class': 10})
clf: RandomForestClassifier
RandomForestClassifier(random_state=42)
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

```
# Get model's predictions
predicted_labels = cleaner.predict(X_train)

# Create an empty list to store DataFrames
suspect_dfs = []

# Loop over the mislabeled indices and create a structured DataFrame for each
for idx in mislabeled_indices:
    # Create a DataFrame for the suspected mislabeled data point
    df_suspect = pd.DataFrame([X_train[idx]], columns=iris.feature_names)
    df_suspect.insert(0, "Index", idx)  # Insert index column
    df_suspect["True Label"] = y_train[idx]  # Correct label
    df_suspect["Previously Assigned Label"] = predicted_labels[idx]  # What it was classified as before

    # Append the current suspect DataFrame to the list
    suspect_dfs.append(df_suspect)

# Combine all the suspect DataFrames into a single DataFrame
df_all_suspects = pd.concat(suspect_dfs, ignore_index=True)

# Print the full table of suspected mislabeled data points
print("\n                                    Suspected Mislabeled Data Points")
print("-------------------------------------------------------------------------------------------------------------------")

print(df_all_suspects.to_string(index=False))
```

```
                                    Suspected Mislabeled Data Points
-------------------------------------------------------------------------------------------------------------------
 Index  sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  True Label  Previously Assigned Label
    54                6.7               3.0                5.0               1.7           1                         2
```

Task 3:

```python
# Load the Iris dataset
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
df['target'] = iris.target
print(df.head())
```

```
   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1               3.5                1.4               0.2
1                4.9               3.0                1.4               0.2
2                4.7               3.2                1.3               0.2
3                4.6               3.1                1.5               0.2
4                5.0               3.6                1.4               0.2

   target
0       0
1       0
2       0
3       0
4       0
```

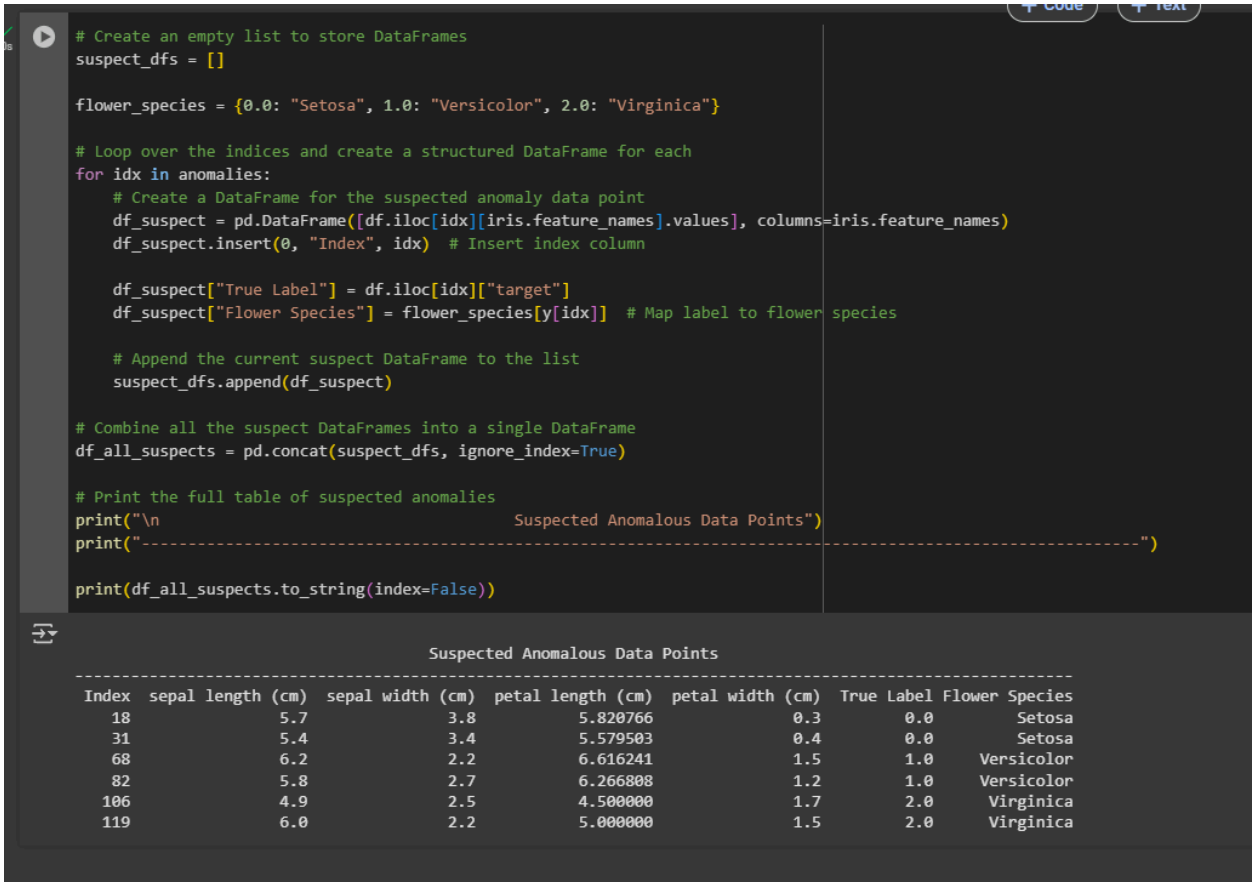
```python
X = iris.data
y = iris.target

# Use CleanLearning for anomaly detection
clf = CleanLearning()
clf.fit(X, y)

# Find potential anomalies in labels
label_issues = clf.find_label_issues(X, y)

# Output the anomalies
anomalies = np.where(label_issues["is_label_issue"])[0]
print(f"Anomalies detected at indices: {anomalies}")
print(f"Suspected anomaly values: {X[anomalies]}")
```

```
Anomalies detected at indices: [ 18  31  68  82 106 119]
Suspected anomaly values: [[5.7        3.8        5.82076585 0.3       ]
 [5.4        3.4        5.57950291 0.4       ]
 [6.2        2.2        6.61624076 1.5       ]
 [5.8        2.7        6.26680751 1.2       ]
 [4.9        2.5        4.5        1.7       ]
 [6.         2.2        5.         1.5       ]]
```



```python
# Create an empty list to store DataFrames
suspect_dfs = []

flower_species = {0.0: "Setosa", 1.0: "Versicolor", 2.0: "Virginica"}

# Loop over the indices and create a structured DataFrame for each
for idx in anomalies:
    # Create a DataFrame for the suspected anomaly data point
    df_suspect = pd.DataFrame([df.iloc[idx][iris.feature_names].values], columns=iris.feature_names)
    df_suspect.insert(0, "Index", idx)  # Insert index column

    df_suspect["True Label"] = df.iloc[idx]["target"]
    df_suspect["Flower Species"] = flower_species[y[idx]]  # Map label to flower species

    # Append the current suspect DataFrame to the list
    suspect_dfs.append(df_suspect)

# Combine all the suspect DataFrames into a single DataFrame
df_all_suspects = pd.concat(suspect_dfs, ignore_index=True)

# Print the full table of suspected anomalies
print("\n                                    Suspected Anomalous Data Points")
print("-------------------------------------------------------------------------------------------------------------------")

print(df_all_suspects.to_string(index=False))
```

```
                                    Suspected Anomalous Data Points
-------------------------------------------------------------------------------------------------------------------
 Index  sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  True Label Flower Species
    18                5.7               3.8           5.820766               0.3         0.0         Setosa
    31                5.4               3.4           5.579503               0.4         0.0         Setosa
    68                6.2               2.2           6.616241               1.5         1.0     Versicolor
    82                5.8               2.7           6.266808               1.2         1.0     Versicolor
   106                4.9               2.5           4.500000               1.7         2.0      Virginica
   119                6.0               2.2           5.000000               1.5         2.0      Virginica
```

https://github.com/RyanJohnV/RyanVarghese_100870665_SoftQuality_Lab5.git