The problem is with the fit function for the MrSEQLClassifier()

Initialize classifier: clf = MrSEQLClassifier()

Fit the classifier to data (**works as expected**) : clf.fit(X_train , y_train)

Fit the classifier to new data (**does not work as expected**): clf.fit(X_train_2 , y_train_2)

What happens when you fit to new data? (Diving into class source code)

The fit method within the MrSEQLClassifier() is called

Problem line is highlighted in red.

This method here states that it 'fits the model according to given training time series data' but it does not do that. It incorrectly retains a memory of previous fitting due to line in red. This is expanded below.

Fit method in MrSEQLClassifier:

```
def fit(self, X, y):
    """
    Fit the model according to the given training time series data.
    Parameters
    ----------
    X : Time series data.
    y : Target vector relative to X.

    Returns
    -------
    self
       Fitted estimator.
    """
    X, y = check_X_y(X,y, coerce_to_numpy=True)

    # transform time series to multiple symbolic representations
    mr_seqs = self._transform_time_series(X)

    self.seql_clf.fit(mr_seqs,y)
    self.classes_ = self.seql_clf.classes_
    self.sequences = self.seql_clf.get_sequence_features()

    # if seql is being used to select features
    # first computing the feature vectors
    # then fit the new data to a logistic regression model
    if self.seql_mode == 'fs':
       train_x = self._to_feature_space(mr_seqs)
       self.ots_clf = LogisticRegression(
          solver='newton-cg', multi_class='multinomial', class_weight='balanced').fit(train_x, y)
       self.classes_ = self.ots_clf.classes_

    self._is_fitted = True
    return self
```

Fit method in seql class, it is used within the MrSEQLClassifier() :

self.seql_clf.fit(mr_seqs,y)

This line fits a 'seql' classifier to the multi resolution representation of the input training data (X).

Seql is itself a class and this is initialized when the MrSEQL model is first created. The init_ method for MrSEQLClassifier has the following code :

self.seql_clf = SEQLCLF() # seql model

When seql is first initialized features and coefficients are set to an empty list.

```
def __init__(self):
    self.features = []
    self.coefficients = []
```

Therefore when the red line is first run it correctly learns coefficients and features from the training data.

However when the fit method is run for the second time, the seql init method does not run. Therefore self.features and self.coefficients are not set to empty lists.

Instead only the seql_clf.fit() method is run.

Looking at the seql_clf.fit() method :

```
def fit(self, mr_seqs, labels):
    self.classes_ = np.unique(labels)
    for ul in self.classes_:
        tmp_labels = [1 if l == ul else -1 for l in labels]
        f,c = self._fit_binary(mr_seqs, tmp_labels)
        self.features.append(f)      → These lines incorrectly append to list of features that have not been wiped
        self.coefficients.append(c)     and are retained from when model was last fitted
        if self.is_binary():
            self.features.append(f)
            self.coefficients.append(self._reverse_coef(c))
            break
```

expanding internal _fit_binary method in blue:

This method does not set the feature and coefficients attributes to empty lists, it creates new features and coefficients variables and returns them.

```
def _fit_binary(self, mr_seqs, labels):
    # labels have to be 1 and -1
    features = []
    coefficients = []
    for rep in mr_seqs:
        m = PySEQL()
        f,c = m.learn(rep, labels)
        features.append(f)
        coefficients.append(c)

    return features, coefficients
```