```python
# Training and testing functions

import pandas as pd
import numpy as np

def fit_model(ticker, name, model, X_train, y_train, X_test, test_output):

    model.fit(X_train, y_train)
    prediction = pd.DataFrame(model.predict(X_test))

    # Calculates hit rate using predicted output and ground truth
    output = (1.0+prediction*test_output)/2.0
    hit_rate = np.mean(output)

    # Drops the initial data value to obtain predictions from day one onward
    hit_rate = hit_rate.drop(0)

    # Prints results
    print(ticker + " " + name + " One-Split Finished")

    return hit_rate

def fit_model_cross_validation(ticker, name, model, X_data, y_data, splits=3):

    from sklearn.model_selection import TimeSeriesSplit

    # Initializes time series split object
    time_series_cv = TimeSeriesSplit(n_splits=splits)

    hit_rate = []
    split_cnt = 1

    # Create time series split indices. Trains and tests
    # model on split data
    for train_index, test_index in time_series_cv.split(X_data):
        print train_index, test_index
        X_train, X_test = X_data[train_index], X_data[test_index]
        y_train, y_test = y_data[train_index], y_data[test_index]
        model.fit(X_train, y_train)
        model_prediction = model.predict(X_test)

        # Calculates hit rate using predicted output and ground truth
        correct_rate = (1.0 + model_prediction * y_test) / 2.0
        mean = np.mean(correct_rate, axis=0)
        hit_rate.append(np.delete(mean, [0], axis=0))

        # Print accuracy
        print(ticker + " " + name + " Cross Valid " + str(split_cnt) + " Finished")

        split_cnt = split_cnt + 1

    return hit_rate

```