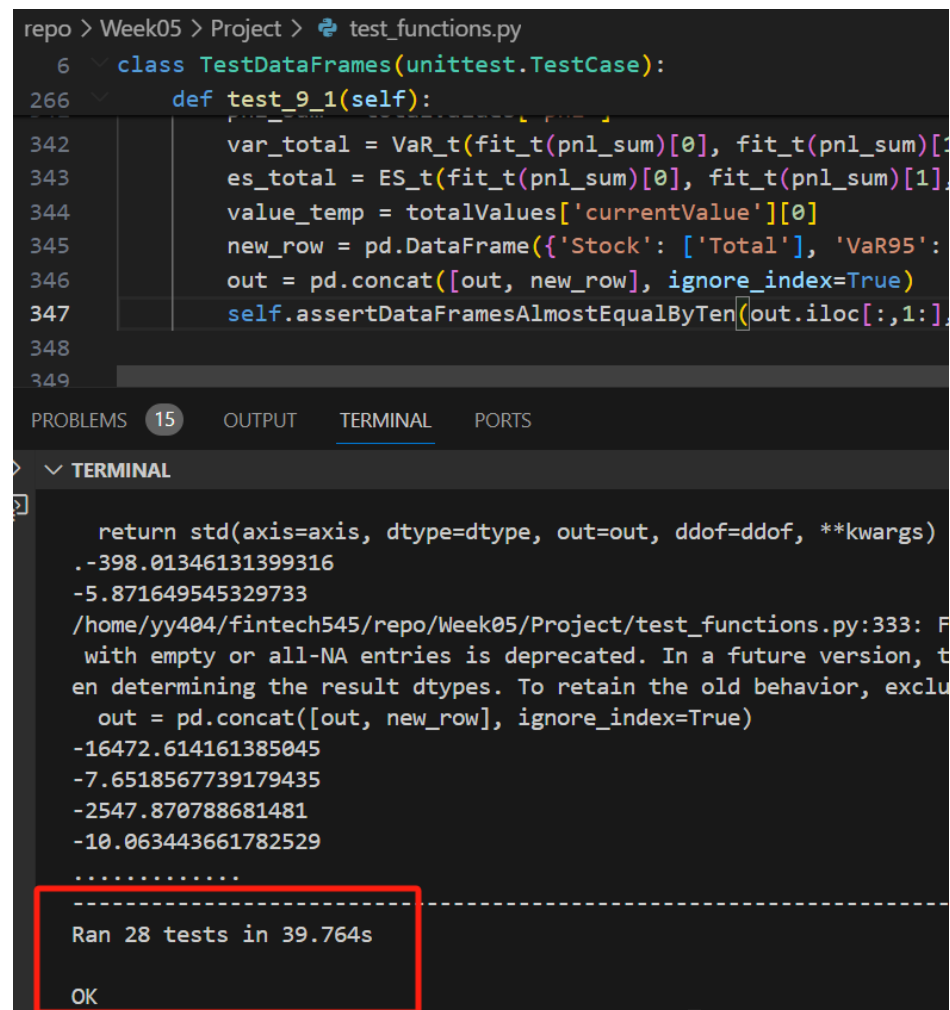


## Project Week05 – Yue Yu(yy404)

### Problem1

Pass all the tests.



The screenshot shows a Jupyter Notebook interface. The top part displays a Python code snippet from a file named `test_functions.py`. The code defines a class `TestDataFrames` that inherits from `unittest.TestCase`. Inside the class, there is a method `test_9_1` which performs several operations: it calculates `var_total` and `es_total` using `VaR_t` and `ES_t` functions, extracts a value from `totalValues`, creates a new row in a pandas DataFrame, and then concatenates it to an existing DataFrame. The method concludes with `self.assertDataFramesAlmostEqualByTen`. Below the code, the 'TERMINAL' tab is active, showing the execution output. The output includes several numerical values, a warning message about deprecated pandas behavior, and a summary line 'Ran 28 tests in 39.764s' which is highlighted with a red rectangle. An 'OK' button is visible at the bottom of the terminal output.

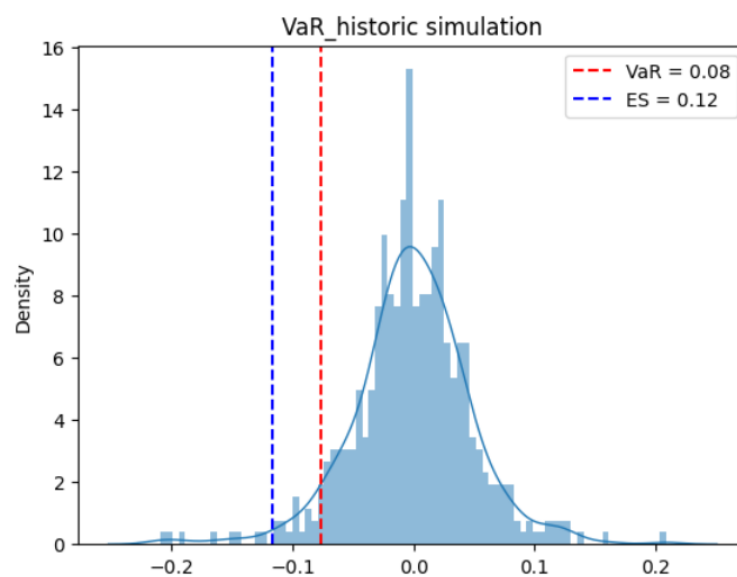
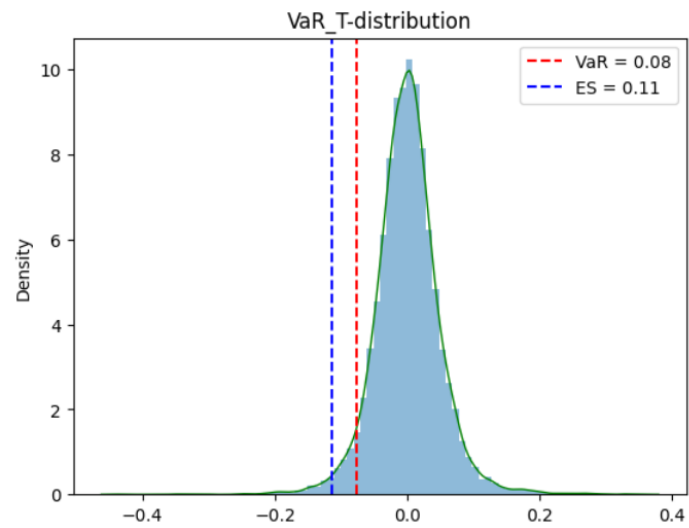
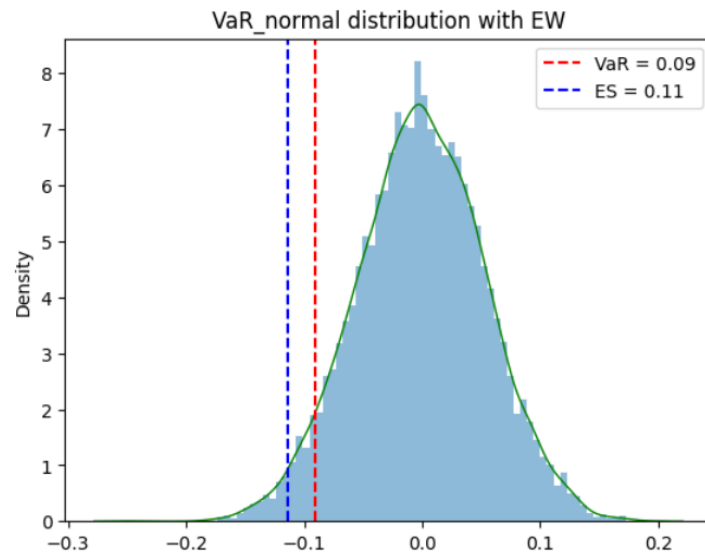
```
repo > Week05 > Project > test_functions.py
6 class TestDataFrames(unittest.TestCase):
266 def test_9_1(self):
342     var_total = VaR_t(fit_t(pnl_sum)[0], fit_t(pnl_sum)[1])
343     es_total = ES_t(fit_t(pnl_sum)[0], fit_t(pnl_sum)[1])
344     value_temp = totalValues['currentValue'][0]
345     new_row = pd.DataFrame({'Stock': ['Total'], 'VaR95':
346     out = pd.concat([out, new_row], ignore_index=True)
347     self.assertDataFramesAlmostEqualByTen(out.iloc[:,1:],
348
349

PROBLEMS 15 OUTPUT TERMINAL PORTS
▼ TERMINAL
return std(axis=axis, dtype=dtype, out=out, ddof=ddof, **kwargs)
.-398.01346131399316
-5.871649545329733
/home/yy404/fintech545/repo/Week05/Project/test_functions.py:333: F
with empty or all-NA entries is deprecated. In a future version, t
en determining the result dtypes. To retain the old behavior, exclu
out = pd.concat([out, new_row], ignore_index=True)
-16472.614161385045
-7.6518567739179435
-2547.870788681481
-10.063443661782529
.....
-----
Ran 28 tests in 39.764s
OK
```

### Problem2

Using the data in problem1.csv, the VaR and ES are calculated by 3 fitting methods as following:

	VaR	ES
Normal distribution with EW	0.0911693	0.1141065
MLE + t distribution	0.0764756	0.1132179
Historic Simulation	0.0759807	0.1167766



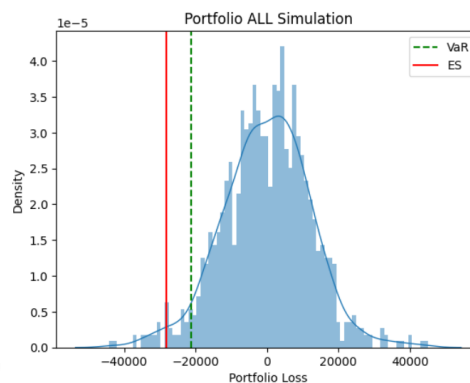
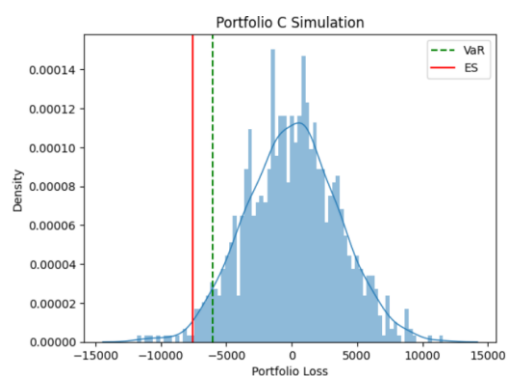
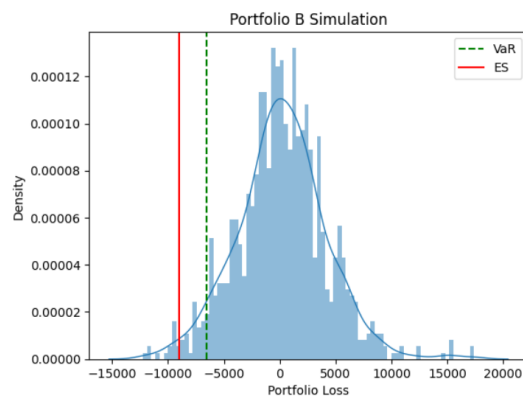
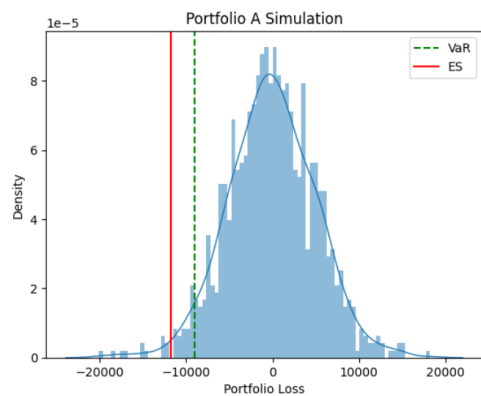
We can see that the VaR under the Normal Distribution with EW is noticeably the largest,

while the VaR under MLE-fitted t distribution and Historical Simulation are approximately similar. However, the ES estimates for these three fitting methods are very close. I believe this is because the Normal Distribution is more concentrated in the center, whereas the t Distribution has thicker tails. Moreover, from the graphical perspective, the t Distribution aligns better with Historical Simulation, and one could argue it is closer to real-world conditions.

## Problem3

By fitting Generalized T model to portfolio A and B, and Normal distribution to portfolio ES, the VaR and ES as following:

Portfolio	VaR	ES
A	9022.89	11752.09
B	6886.91	8962.14
C	6217.31	7568.13
Total	21333.03	28153.62



By comparing the VaR from Copula with the methods last week, we can see the VaR is

completely different, which due to the different portfolios and returns.

<b>Portfolio</b>	<b>Copula</b>	<b>Delta Normal</b>	<b>Monte Carlo</b>	<b>Historic</b>
<b>A</b>	9022.89	15426.97	14014.13	17933.41
<b>B</b>	6886.91	8082.57	7474.11	10983.46
<b>C</b>	6217.31	18163.29	16285.41	21409.75
<b>Total</b>	21333.03	38941.38	35642.05	49544.19