

# Applying a Random Forest II

MGMT 638: Data-Driven Investments: Equity

Kerry Back, Rice University



## Outline

- Read current data
- Interact features with market volatility
- Load saved model
- Make predictions



Read data



```
In [36]: import pandas as pd  
df = pd.read_excel("data-current-2023-11-13.xlsx")
```



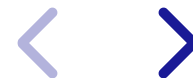
Read model



```
In [37]: # Local machine version  
  
from joblib import load  
forest = load("forest_ver2.joblib")
```

In [ ]: *# colab version*

```
from joblib import load
from google.colab import drive
drive.mount('/content/drive')
forest = load('/content/drive/My Drive/forest_ver2.joblib')
```



Make predictions and save





```
In [38]: features = [
        "marketcap",
        "pb",
        "mom",
        "volume",
        "volatility",
        "roe",
        "accruals",
        "agr"
    ]
    features.sort()

    for x in features:
        df[x+"_vol"] = df[x]*df.mktvol
    features_final = features + [x+"_vol" for x in features]

    df["predict"] = forest.predict(X=df[features_final])
    df.to_excel("predict-2023-11-13.xlsx")
```

Distribution of predictions



```
In [39]: df.predict.describe()
```

```
Out[39]: count    1753.000000  
mean         50.043001  
std          1.615579  
min          38.642801  
25%          50.023600  
50%          50.649518  
75%          50.925365  
max          53.033142  
Name: predict, dtype: float64
```



Create an interactive predictor



```
In [40]: import numpy as np

def predict(mktvol):
    lst = []
    for x in features:
        item = input(f"Input {x}: ")
        lst.append(float(item))
    lst = lst + [mktvol*x for x in lst]
    arr = np.array(lst).reshape(1, len(lst))
    d = pd.DataFrame(arr, columns=features+[x+"_vol" for x in features])
    return forest.predict(d).item()
```

Use the interactive predictor



In [ ]: `predict(0.15)`

