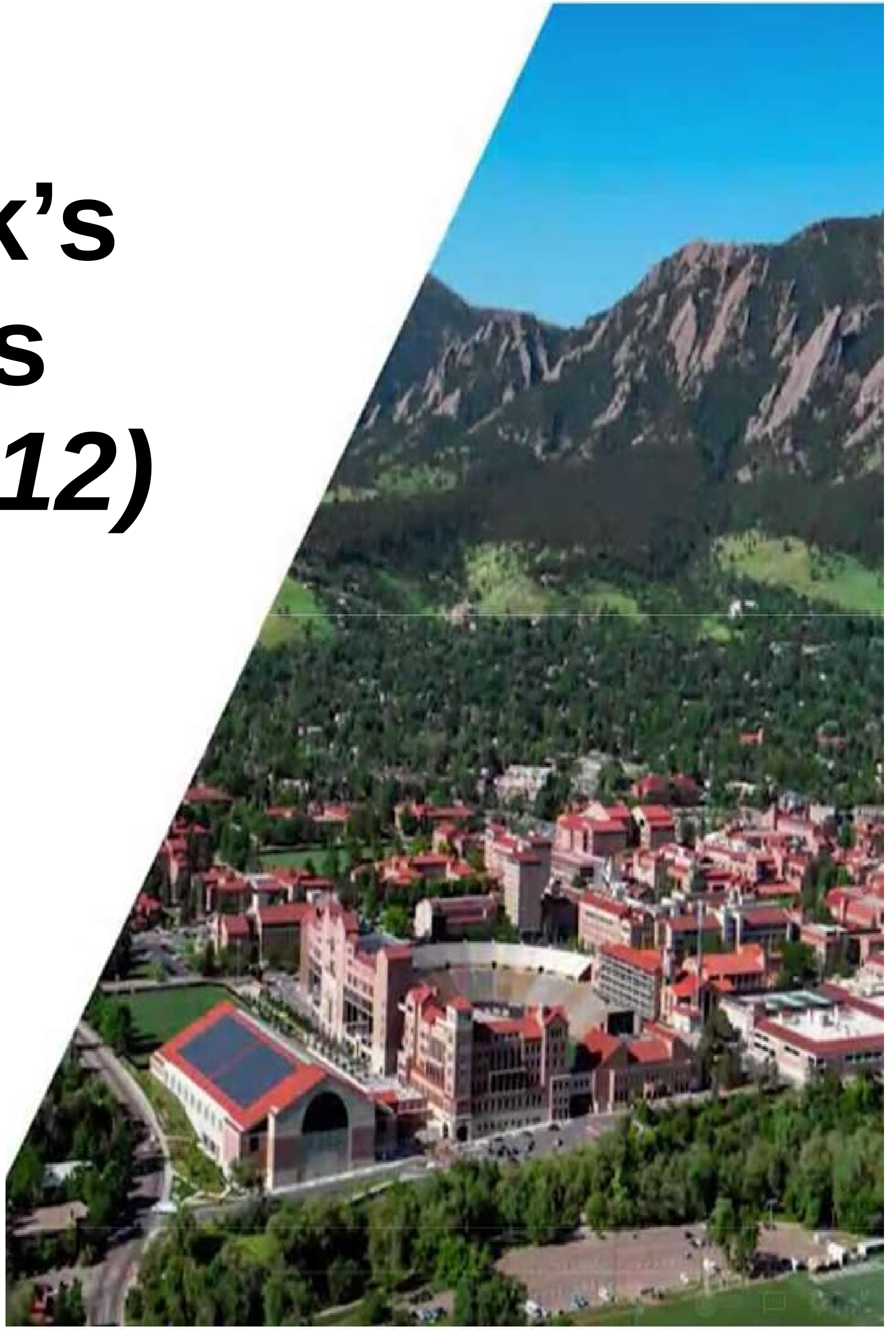
Modeling Stock's Closing Prices
by XX (2025-03-12)



Master of Science in Data Science
UNIVERSITY OF COLORADO **BOULDER**

Table of contents

- Introduction:
  - Problem Statement
  - Data Set
  - Tools & Techniques
- Related Work:
  - Description
  - Data Sources & Pre-processing
  - Tools & Techniques
- Models & Analysis:
  - Single-Predictor Linear Regression
  - Multi-Predictor Linear Regression
  - Single Decision Tree
  - Random Forest
  - Ada Boost on Random Forest
  - Gradient Boost
  - Xtreme Boost (XGB)
- Summary and Conlusions: Stacking estimators
- Future Work
- References

Data Science
UNIVERSITY OF COLORADO BOULDER

Caution: Don’t Try This at Home

- Investments involve risk. You may loose some or all your money.
- This is an academic paper. It is not investment advise.

Introduction: Problem Statement

- Stock prices move at every second. Having a feel of direction and magnitude of the moves is very useful.
- Predict closing price of a stock (S&P500) nperiods forward.
- Main goal: predict closing price of S&P500 constituent based on available time series of open, high, low, volume, and close prices of other securities, indices and commodities, and open, high, low, and volume of the security itself. Each security brings 4 or 5 predictors: open, high, low, close prices, and volume, with volume some times not (completely) available. This makes the total number of possible predictors a number between 2,000 and 2,500.
- All possible combinations of predictors 8.75 10^18 years, not an option.
- 500 securities on S&P500 only scratching the surface.
- Approach: incremental in complexity. Single predictor linear regression, multi-predictor linear models, decision trees (DT), random forest (RF), Adaptive Boosting (Ada Boost), Gradient Boosting (GB), and Extreme Gradient Boost (XGB).
- Stack / average output, and predict 500 securities (out-of-sample data), calculate and estimate prediction error of the model.
- AAPL as example to build and tune the model.

Data Set: Time series (yahoo.f)

Date	Open	High	Low	Close	Volume
2025-02-04	5998.14013671875	6042.47998046875	5990.8701171875	6037.8798828125	4410160000
2025-02-05	6020.4501953125	6062.85986328125	6007.06005859375	6061.47998046875	4756250000
2025-02-06	6072.22021484375	6084.02978515625	6046.830078125	6083.56982421875	4847120000
2025-02-07	6083.1298828125	6101.27978515625	6019.9599609375	6025.990234375	4766900000
2025-02-10	6046.39990234375	6073.3798828125	6044.83984375	6066.43994140625	4458760000
2025-02-11	6049.31982421875	6076.27978515625	6042.33984375	6068.5	4324880000
2025-02-12	6025.080078125	6063	6003	6051.97021484375	4627960000
2025-02-13	6060.58984375	6116.91015625	6050.9501953125	6115.06982421875	4763800000
2025-02-14	6115.52001953125	6127.47021484375	6107.6201171875	6114.6298828125	4335190000
2025-02-18	6121.60009765625	6129.6298828125	6099.509765625	6129.580078125	4684980000
2025-02-19	6117.759765625	6147.43017578125	6111.14990234375	6144.14990234375	4562330000
2025-02-20	6134.5	6134.5	6084.58984375	6117.52001953125	4813690000

- Securities: S&P 500, Gold, Bonds, Crude Oil, Indexes (US, world).
  - Open, High, Low, Close, Volume (*) , daily.
- Trading days on S&P 500.
- Daily versus nperiods. Securities trade in different timelines. Generally clean, complete, reliable, scalable.
- Training/Testing set: train and tune with 1 stock, error measure with all others.

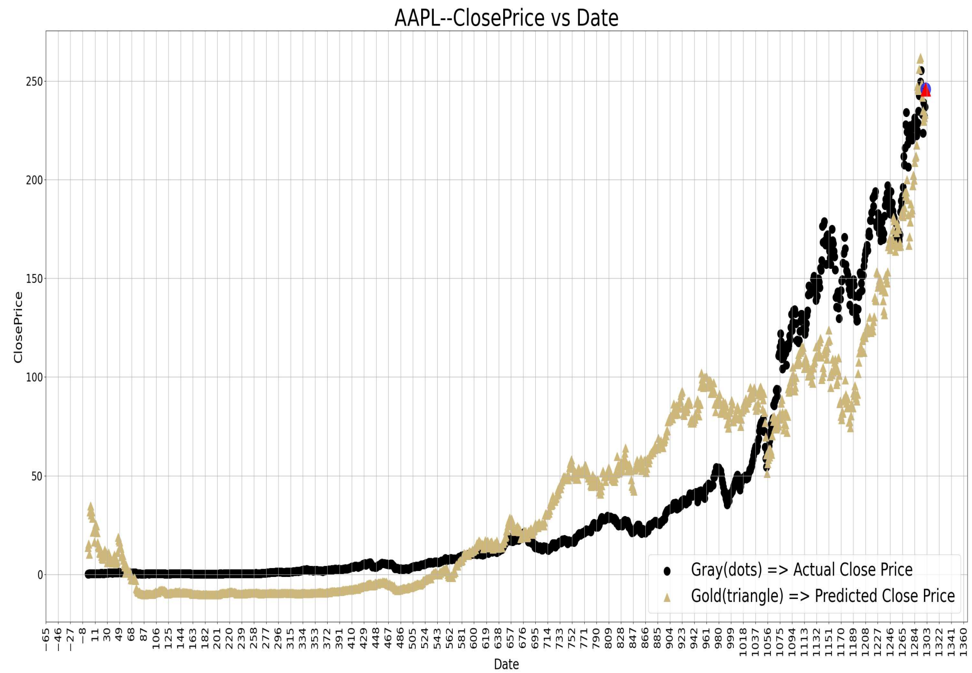
Data Science
UNIVERSITY OF COLORADO BOULDER

Tools & Techniques

- Data is numeric, continuous and sequential (ordered). Time series.
- Py: statsmodels, scipy.stats, sklearn, matplotlib, yfinance.
- Stats, regression, trees, random forest, ensemble.

Single-Predictor linear regression

```
Line 1133:
lowestErrorSPModel                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:              AAPLClose   No. Observations:                 1301
Model:                            GLM   Df Residuals:                     1299
Model Family:                Gaussian   Df Model:                            1
Link Function:               Identity   Scale:                          667.84
Method:                          IRLS   Log-Likelihood:                -6075.9
Date:                Tue, 04 Mar 2025   Deviance:                   8.6753e+05
Time:                        10:49:39   Pearson chi2:                 8.68e+05
No. Iterations:                     3   Pseudo R-squ. (CS):             0.9896
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept    -10.7748      0.982    -10.975      0.000     -12.699      -8.851
BKNGHigh       0.0511      0.001     77.069      0.000       0.050       0.052
==============================================================================
lowestTestErrorPct= 0.0577 %
Number of models= 2384
testActualClose= 245.83
meanTestPredSP= 141.86 across all models all models.
meanTestErrorPctSP= -42.2926 % ; mean test error across all models.
meanRMSETrainErrorPctSP= 13.28 %

Total Time:  2376.8 secs.
```

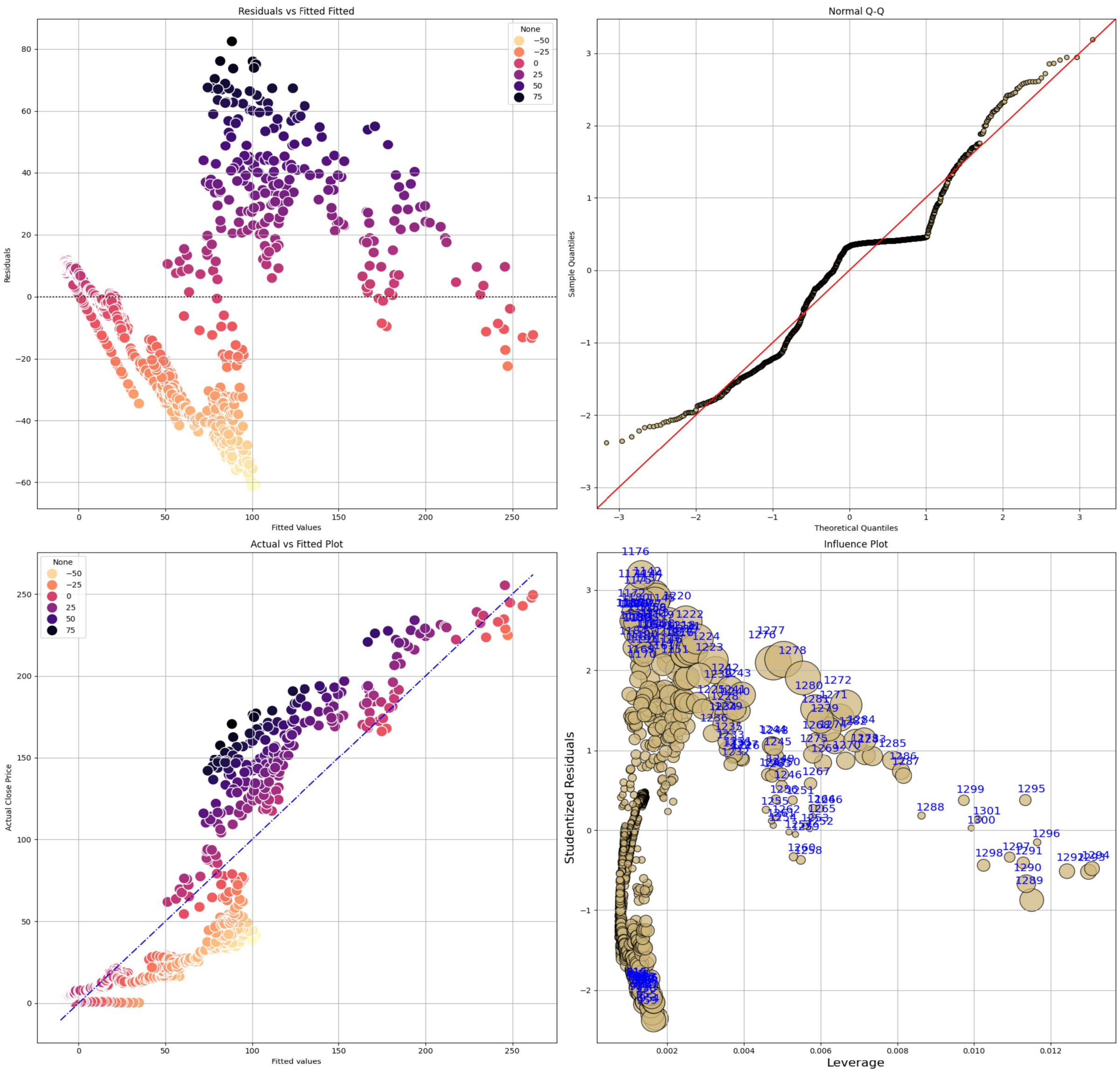


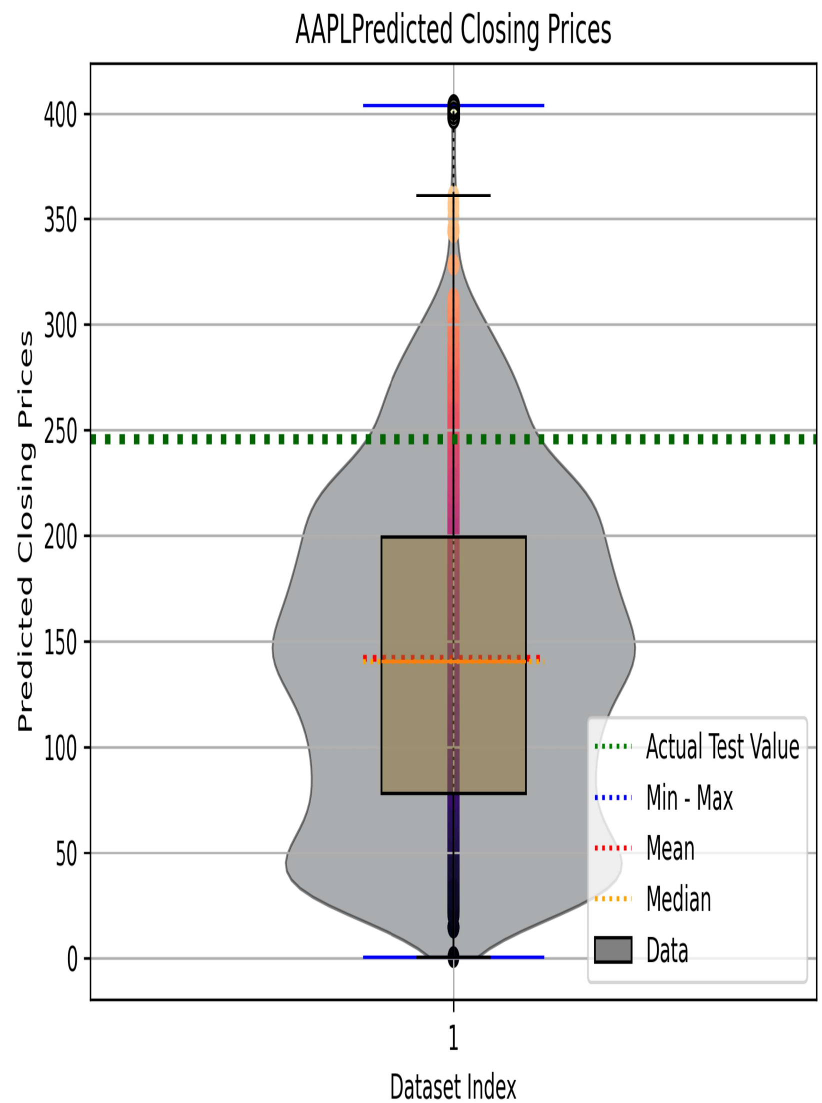
- Lowest error model: error 0.06%, predicted close price 245.69 vs actual 245.83. (AAPL, 2025-02-20)

- Average prediction: 141.86, with 42.29% (across 2384 models)

Data Science
UNIVERSITY OF COLORADO BOULDER

Single-Predictor linear regression



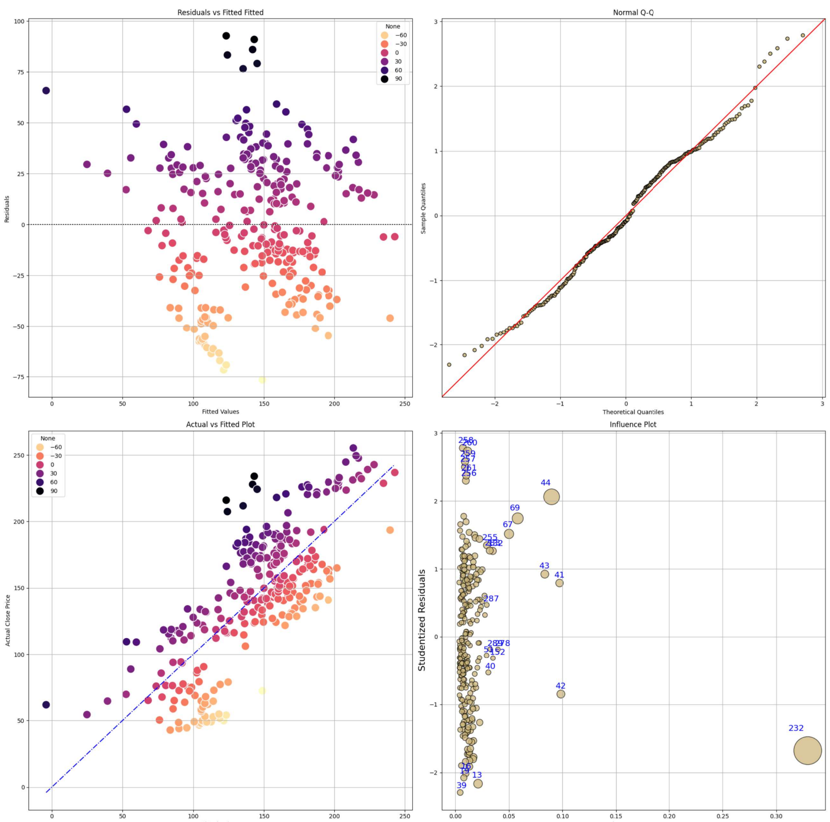


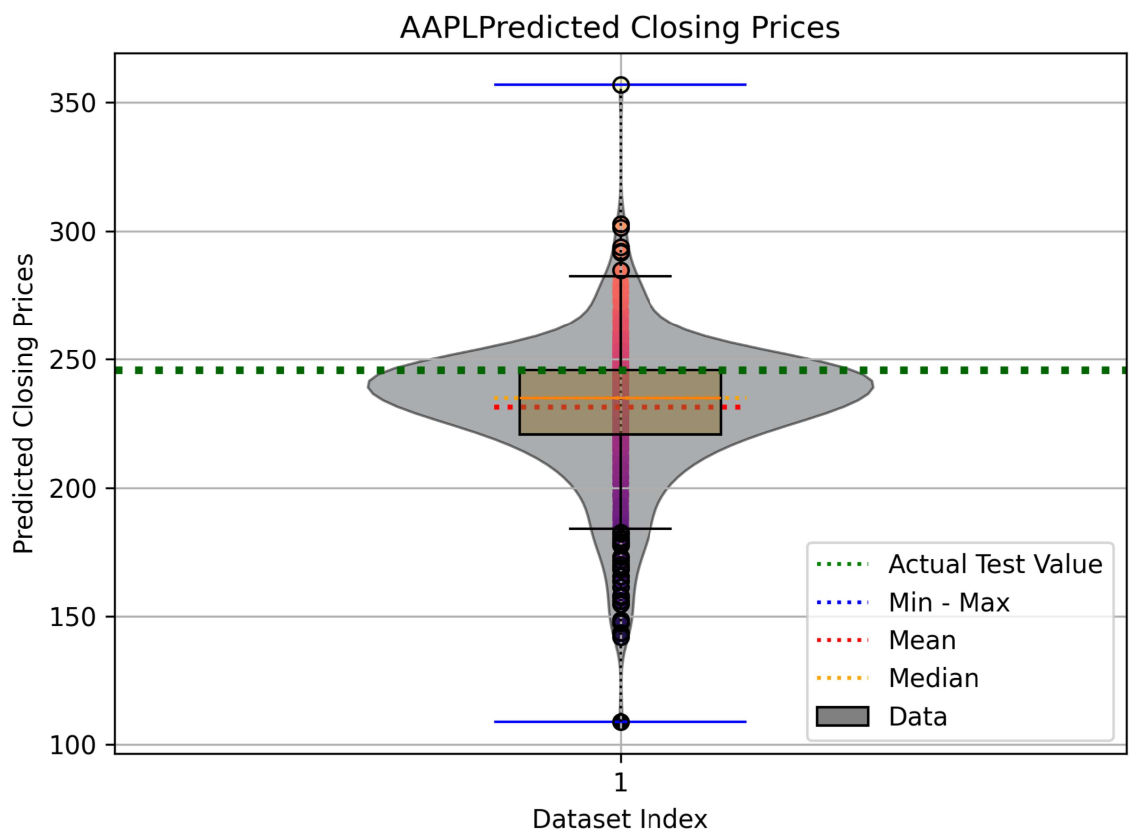
Data Science
UNIVERSITY OF COLORADO BOULDER

Multi-Predictor linear regression

```
Line 1219:
lowestModel=                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:              AAPLClose   No. Observations:                  289
Model:                            GLM   Df Residuals:                      285
Model Family:                Gaussian   Df Model:                            3
Link Function:               Identity   Scale:                          1120.5
Method:                          IRLS   Log-Likelihood:                -1422.7
Date:                Thu, 06 Mar 2025   Deviance:                   3.1933e+05
Time:                        06:37:20   Pearson chi2:                 3.19e+05
No. Iterations:                     3   Pseudo R-squ. (CS):             0.7487
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   -104.0865     23.180     -4.490      0.000    -149.519     -58.654
AAPLVolume -3.888e-07   4.37e-08     -8.899      0.000   -4.74e-07   -3.03e-07
AEEOpen        3.5373      0.276     12.834      0.000       2.997       4.078
AEEVolume    1.17e-05   3.01e-06      3.883      0.000    5.79e-06    1.76e-05
==============================================================================
```

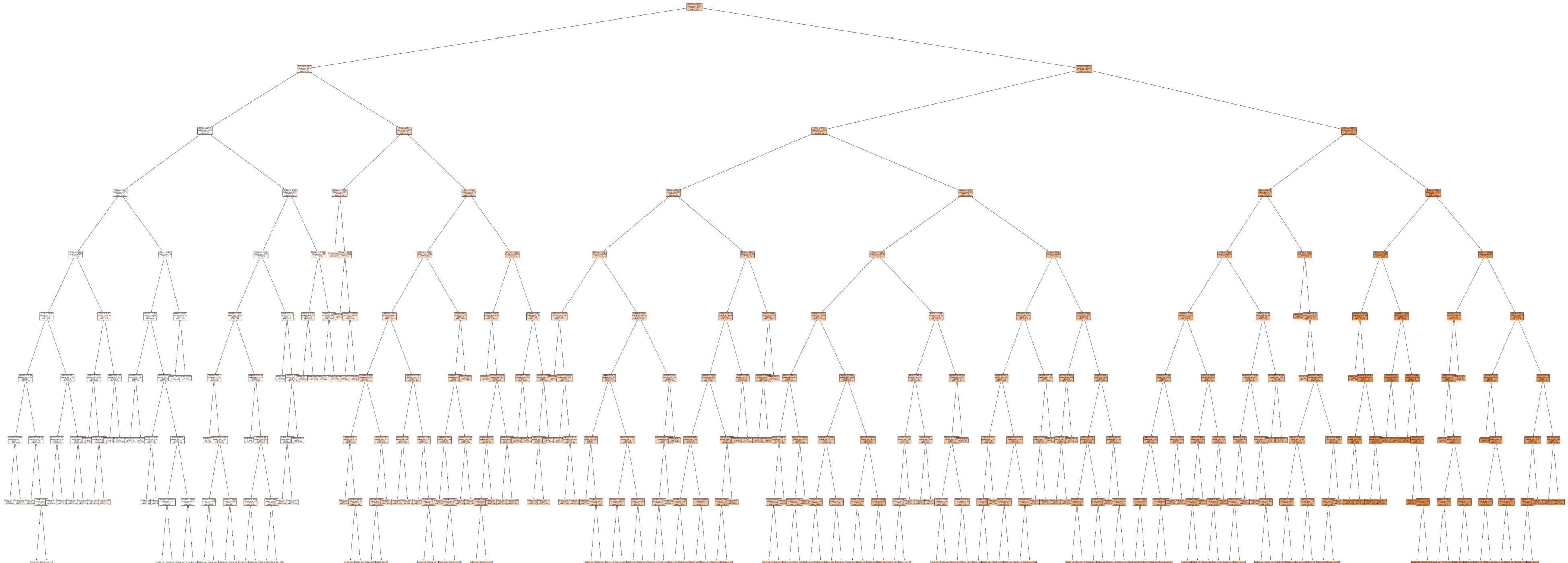
```
lowestPredError%= 0.0105 '%'on test set.
lowestRMSEPct= 13.52  '%' average error on the training set.
closeActual= 245.83  the actual closing Price on test date. 2025-02-20
lowestClosePred = 245.8  the predicted closing Price on test date.

Line 1306: This predictor= AAPLVolume R2= 0.21826264148013774 vif= 1.279202009602554 is good on vif.

Line 1306: This predictor= AEEOpen R2= 0.18640760268860057 vif= 1.2291166968922078 is good on vif.

Line 1306: This predictor= AEEVolume R2= 0.07757293239640073 vif= 1.0840965482484484 is good on vif.

Line 1232: nModels= 481 samples of random securities= 20
meanPredError= 8.38 '%'
meanRMSEPct= 8.52 '%'
meanClosePred= 231.27

Line 1304: dwStat= 0.35299861477533556

Line 1311: time= 14065.0[redacted]
```





Single Decision Tree

```
Line 337: yTest [245.83000183]

Line 1336: DecisionTreeRegressor object complete.

Line 1359 Best Hyperparameters: {'criterion': 'friedman_mse', 'max_depth': 9, 'max_features': 'log2', 'min_samples_leaf': 1, 'min_samples_split': 2}
bestScore= 2.8068055606009397

Line 1362: DecisionTreeRegressor fit complete.


Line 1365:
dtPred= [244.73092651]

Line 1373:
dtrMSE= 1.21 dtrRMSE= 1.1
dtrRMSEPct= 0.45 '%'

Line 1388: time= 13.0 secs.
```



Data Science
UNIVERSITY OF COLORADO BOULDER

Random Forest

```
Line 1423 Random Forest Best Hyperparameters: {'bootstrap': False, 'criterion': 'absolute_error', 'max_depth': None, 'max_features': 'sqrt', 'min
_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 32}
bestScore= 1056.3332014764514

Line 1444: RandomForestRegressor Tuned fit complete.


Line 1446:
rfrTunedPred= [236.9319458]

Line 1456:
rfrTunedMSE= 79.18
rfrTunedRMSE= 8.9
rfrTunedRMSEPct= 3.62 %
rfrTestErrorPct= -3.62 %

Line 1475:
TargetStock: AAPL
Random Forest time= 32.0 secs.
```

```
paramGridrfr =
    "n_estimators"  32  # error 3.62%
    "criterion"  "absolute_error"  #["squared_error", "friedman_mse", "absolute_error", "poisson"],
    "max_depth"  None  #np.arange(1,10,1), #list(np.arange(1,10,1)),# 30, 40, 50], best is 4
    "min_samples_split"  2  #np.arange(1,20,1), #[None,2,3,4,5,6,7,8,9,10],# 10, 20], best is 4
    "min_samples_leaf"  1  #np.arange(1,10,1),
    "max_features"  "sqrt"  #[1,"log2", "sqrt"],
    "bootstrap"  False  # True, # No bootstrapping: the whole data set is used in each tree.
    # Max sample: this parameter controls the size of the sample for each tree. Default=whole dataset is used.
```

Data Science
UNIVERSITY OF COLORADO BOULDER

Ada Boost on Random Forest

```
Line 1511 AdaBoostRandom Forest Best Hyperparameters: {'learning_rate': 1.0, 'n_estimators': 50}
bestScore= 1078.3308170483674

Line 1532: Ada Boost RandomForestRegressor Tuned fit complete.


Line 1534:
adarfrTunedPred= [238.85747337]

Line 1538:
adarfrTunedMSE= 48.62
adarfrTunedRMSE= 6.97
adarfrTunedRMSEPct= 2.84 %
adarfrTestErrorPct= -2.84 %

Line 1553:
TargetStock: AAPL
Ada Boost Random Forest time= 171.0 secs.[redacted]
```

```
paramGridadarfr =
    "n_estimators"   50 100 200
    #"estimator__max_depth": [None, 1,3,5],
    "learning_rate"   0.01  0.1  1.0
```

Gradient Boost

```
Line 1590 Gradient Boost Hyperparameters: {'learning_rate': 0.2, 'max_depth': None, 'max_features': 'log2', 'min_samples_leaf': 1, 'min_samples_split': 2,
_estimators': 110, 'subsample': 1}
bestScore= 3.412512502659875

Line 1604: Gradient Boost Regressor Tuned fit complete.


Line 1606:
gradBoostTunedPred= [238.23710476]

Line 1617:
gradBoostTunedMSE= 57.65
gradBoostTunedRMSE= 7.59
gradBoostTunedRMSEPct= 3.09 %
gradBoostTestErrorPct= -3.09 %

Line 1623:
TargetStock: AAPL
Gradient Boost Regressor time= 17.0 secs.
```

```
paramGridGradBoost =

    'n_estimators'   110   #[90,100,110,120], # 300, 400], #200 3.09%; 120 3.09%

    'learning_rate'   0.2   # [0.01, 0.1, 0.2], #0.1

    'max_depth'   None   #[3, 4, 5], #4

    'min_samples_split'   2   #[2, 5, 10], #5

    'min_samples_leaf'   1   #[1, 2, 4], #2

    'subsample'   1   #[0.8, 0.9, 1.0],

    'max_features'   'log2'  #[1, 'sqrt', 'log2'] #['auto', 'sqrt', 'log2'] #sqrt
```

Extreme Gradient Boost (XGB)

```
Line 1663 XGB Hyperparameters: {'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': None, 'min_child_weight': 3, 'n_estimators': 80, 'reg_alpha': 0, 'reg_lambda': 1
, 'subsample': 1}
bestScore= 1180.7493688552192

Line 1677: XGB Regressor Tuned fit complete.


Line 1679:
XGBTunedPred= [240.70522]

Line 1690:
XgradBoostTunedMSE= 26.26
XgradBoostTunedRMSE= 5.12
XgradBoostTunedRMSEPct= 2.08 %
XgradBoostTestErrorPct= -2.08 %

Line 1697:
TargetStock: AAPL
XGB Regressor time= 130.0 secs.
```

```
paramGridXGradBoost =
    'n_estimators'   80   #[100,200,300], # Higher number of estimators reduces error/increases computational time.
    'learning_rate'   0.1   #[0.01, 0.1, 0.3], # Lower rates lead to better generalizations.
    'max_depth'   None   #[3, 4, 5, 6], # The depth of the trees, typically between 2 and 10.
    'min_child_weight'   3   #[1, 3, 5],  #
    'subsample'   1   #[0.7, 0.8, 0.9], # Fraction of samples to train the tree. I will 1 to use all the time series.
    'colsample_bytree'         0.1 0.2 0.3 0.4 0.5 0.6 0.7  0.8  0.9   # Fraction of features (predictors).
    # Add regularization parameters:
    'reg_alpha'   0   #[0, 0.1, 0.5], # L1 Lasso regularization.
    #Adds a penalty term proportional to the value of the coefficients.
    'reg_lambda'   1  #[0.1, 1, 5] #L2 Ridge regularization.
    #Penalty to the squared value of the coefficients.
```

Stacking

```
Line 1607: llModelsDF:
    ModelName  meanPrediction  MeanTestErrorPct
0       DTree      244.730000           -0.4471
1          RF      236.930000            3.6196
2       adaRF      238.860000            2.8363
3   gradBoost      238.240000            3.0887
4         XGB      240.710007            2.0847
meanPredAllModels= 239.89
meanErrorAllModels= -2.42 %
```

Out-of-sample results

- Apply the model to 500 securities on S&P500.
- Estimated error across all unseen test data in S&P500: error -0.71%.
- This means that the stack of 5 models we have just put together, underpredicts closing prices by -0.71%, on average.
- While improvements arrive, I will correct estimates with this mean error accordingly.
- For APPL for example, the prediction is 239.89. Corrected (upwards) by 0.71% becomes 241.59 with a final error estimate of -1.72%.

```
Line 1663: Final Results
Close Price Prediction on 2025-02-20 for 500 S&P500 securities
==============================================================
     Ticker  ActualClose    MeanPred  MeanErrorPct
0       MOS    26.620001   26.782000      0.608561
1       HPE    21.740000   21.394000     -1.591535
2        KO    70.040001   66.652000     -4.837237
3       IQV   194.009995  211.714001      9.125307
4      PARA    11.470000   10.864000     -5.283350
..      ...          ...         ...           ...
378      PM   151.570007  137.345999     -9.384448
379     PNC   191.889999  196.861999      2.591068
380     PNR    95.250000  100.053999      5.043569
381     PNW    90.769997   87.912000     -3.148613
382    PODD   288.290009  269.561997     -6.496240

[383 rows x 4 columns]
meanErrorPct= -0.71 %
```

Final Remarks and Conclusions

Close Price Prediction on 2025-02-20 for 500 S&P500 securities

	Ticker	ActualClose	MeanPred	MeanErrorPct	CorrectedClosePred	CorrectedErrorPct
0	MOS	26.620001	26.782000	0.608561	26.968840	1.306195
1	HPE	21.740000	21.394000	-1.591535	21.543252	-0.893902
2	KO	70.040001	66.652000	-4.837237	67.116987	-4.139604
3	IQV	194.009995	211.714001	9.125307	213.190988	9.822940
4	PARA	11.470000	10.864000	-5.283350	10.939791	-4.585717
..	...	...	...	...	...	...
403	ROP	581.419983	557.724000	-4.075536	561.614867	-3.377903
404	ROST	139.089996	146.738001	5.498602	147.761694	6.196235
405	RSG	230.860001	217.527999	-5.774929	219.045546	-5.077296
406	RTX	125.110001	120.234000	-3.897371	121.072792	-3.199738
407	RVTY	114.720001	114.402000	-0.277197	115.200107	0.420436

[408 rows x 6 columns]
meanErrorPct= -0.7 %

- ➢ single and multi-predictor regression models: “quick” and easy but computationally expensive.
- ➢ Decision trees, random forest and ensemble methods more suitable solutions for high dimensional data.
- ➢ Ensemble stacking (average) -0.7% prediction error.

Future Work

- Single and multi-predictor models: largest share of the coding effort; expensive.
- Decision trees, random forest, and ensemble techniques: efficient/optimized library functions.
- Time-series specific libraries may help manage code length and efficiency.
- Solution based on models built and tuned on one stock (AAPL). Cleaner solution: tune hyperparameters to each specific security. Better quality and accuracy of results.
- S&P500 constituents securities < 10% of total US market. Extend to all US and worldwide.

References

[1] https://finance.yahoo.com/quote/%5EGSPC/

[2] https://pypi.org/project/yfinance/

[3] https://en.m.wikipedia.org/wiki/List_of_S%26P_500_companies

[4] Decision Tree Regressor from sklearn.tree: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

[5] Grid Search Cross Validation: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

[6] Cross Validation cv parameter for time-series: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.TimeSeriesSplit.html

[7] Random Forest Regressor: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[8] Adaptive Boosting (ADA): https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html

[9] Gradient Boosting (GB):
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor.html

[10]Xtreme Gradient Boosting XGB:
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.htmlXtreme

[11] Stacking Regressor:
https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingRegressor.html

[12] Statsmodels GLM:
https://www.statsmodels.org/stable/glm.html

[13] https://www.geeksforgeeks.org/residual-leverage-plot-regression-diagnostic/

[14] https://stackoverflow.com/questions/66493682/glm-residual-in-python-statsmodel

Thank you !