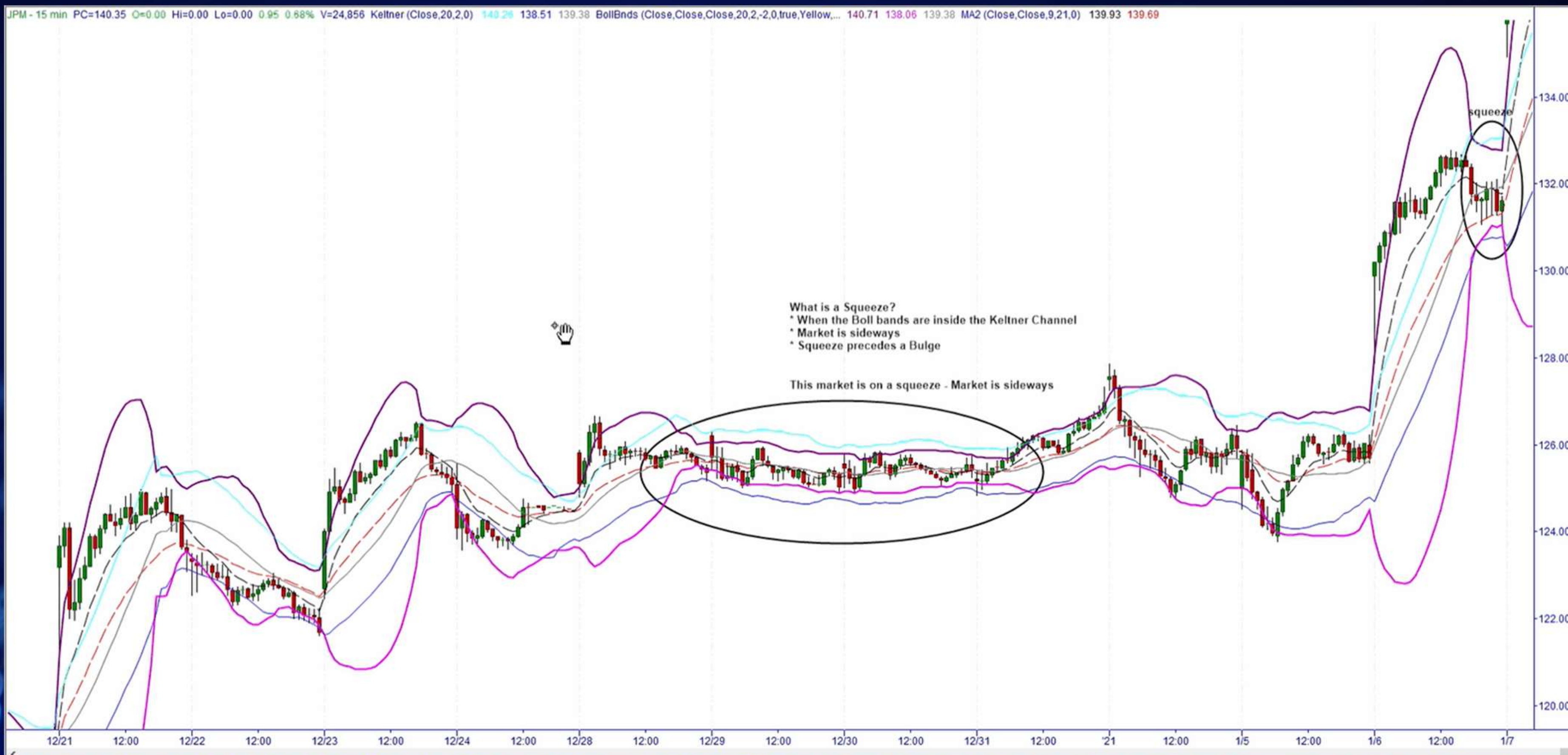
algo-owls

Team Members:
Christian –
Underlying strategy
Jonathan –
Project Plan
Carolina –
Random Forest Metrics
Mark –
LSTM Metrics

January 16th 2021

The Strategy



We aim at anticipating market moves by identifying squeezes that precedes them.
We aim at using 2 EMAs crossovers to anticipate their crossovers for direction predictions.

Project Objectives

REQUIREMENTS

- Build one Machine Learning Model to predict the squeeze breakout and breakout direction.

- Use at least two of the libraries specified below:

Project 2 Requirements

01 Find a FinTech problem that machine learning can help solve.

02 Apply ML in the context of technologies learned.

03 You must use: Scikit-Learn and/or another machine learning library.

04 You must use at least two of the below:

Scikit-Learn	Google Colab
Tensorflow	Amazon SageMaker
Keras	Amazon Lex

3

OUR SCOPE OF WORK

Project Objectives

REQUIREMENTS

- Build one Machine Learning Model to predict the squeeze breakout and breakout direction.

- Use at least two of the libraries specified below:

Project 2 Requirements

01 Find a FinTech problem that machine learning can help solve.

02 Apply ML in the context of technologies learned.

03 You must use: Scikit-Learn and/or another machine learning library.

04 You must use at least two of the below:

Scikit-Learn	Google Colab
Tensorflow	Amazon SageMaker
Keras	Amazon Lex

3

OUR SCOPE OF WORK

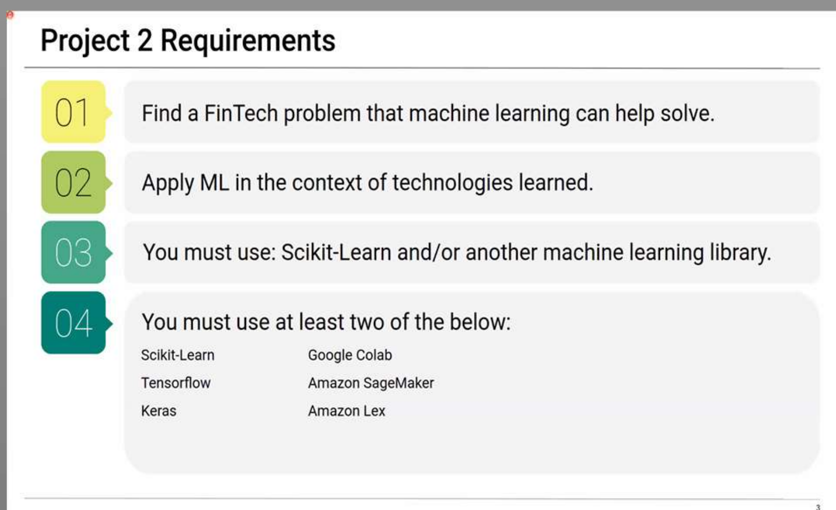
- Build a Machine Learning system comprised of 4 Machine Learning Models:
  - 2 Random Forests Classifiers (squeeze and EMAs crossover)
  - 1 LSTM Binary
  - 1 LSTM price predictor (price entry)

Project Objectives

REQUIREMENTS

- Build one Machine Learning Model to predict the squeeze breakout and breakout direction.

- Use at least two of the libraries specified below:



Project 2 Requirements

01 Find a FinTech problem that machine learning can help solve.

02 Apply ML in the context of technologies learned.

03 You must use: Scikit-Learn and/or another machine learning library.

04 You must use at least two of the below:

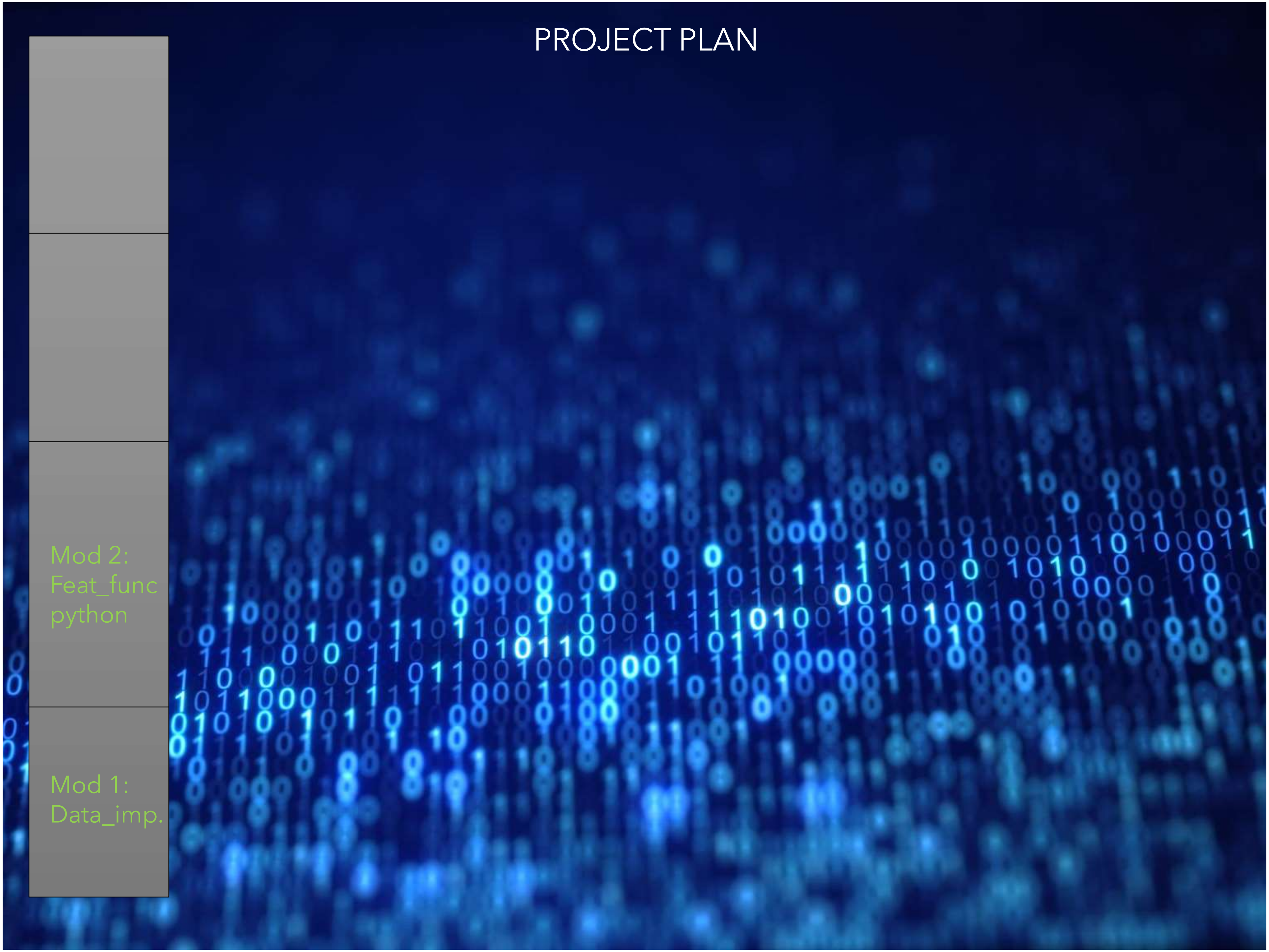
Scikit-Learn	Google Colab
Tensorflow	Amazon SageMaker
Keras	Amazon Lex

OUR SCOPE OF WORK

- Build a Machine Learning system comprised of 4 Machine Learning Models:
  - 2 Random Forests Classifiers (squeeze and EMAs crossover)
  - 1 LSTM Binary
  - 1 LSTM price predictor (price entry)

- Analyze each ML Performance with analytics and metrics.

- Libraries and applications:
  - Scikit-Learn
  - TensorFlow
  - Keras


PROJECT PLAN
Mod 2:
Feat_func
python
Mod 1:
Data_imp.

PROJECT PLAN

Mod 3:
Feat_func
Python 2

Mod 2:
Feat_func
python

Mod 1:
Data_imp.

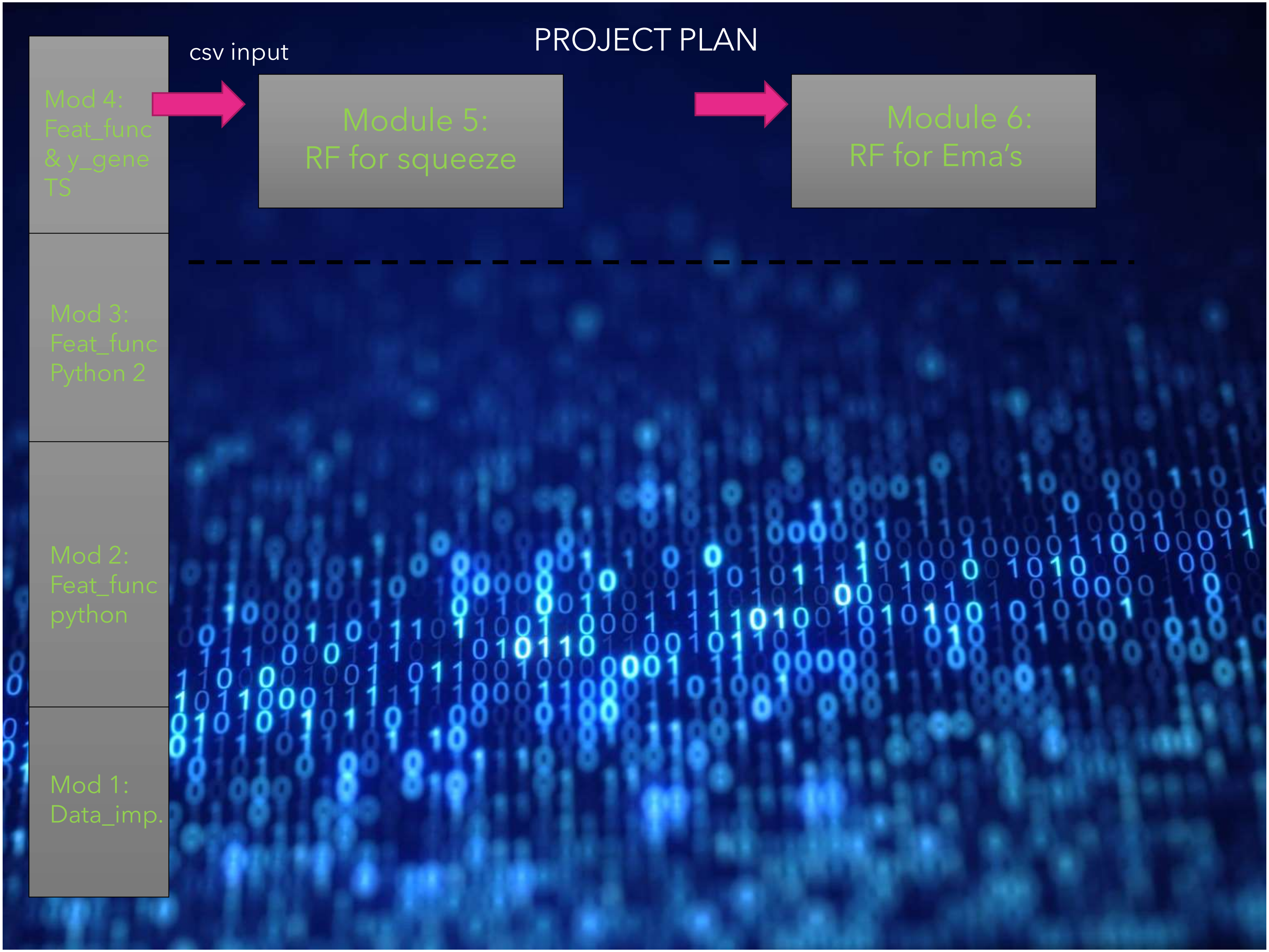
PROJECT PLAN

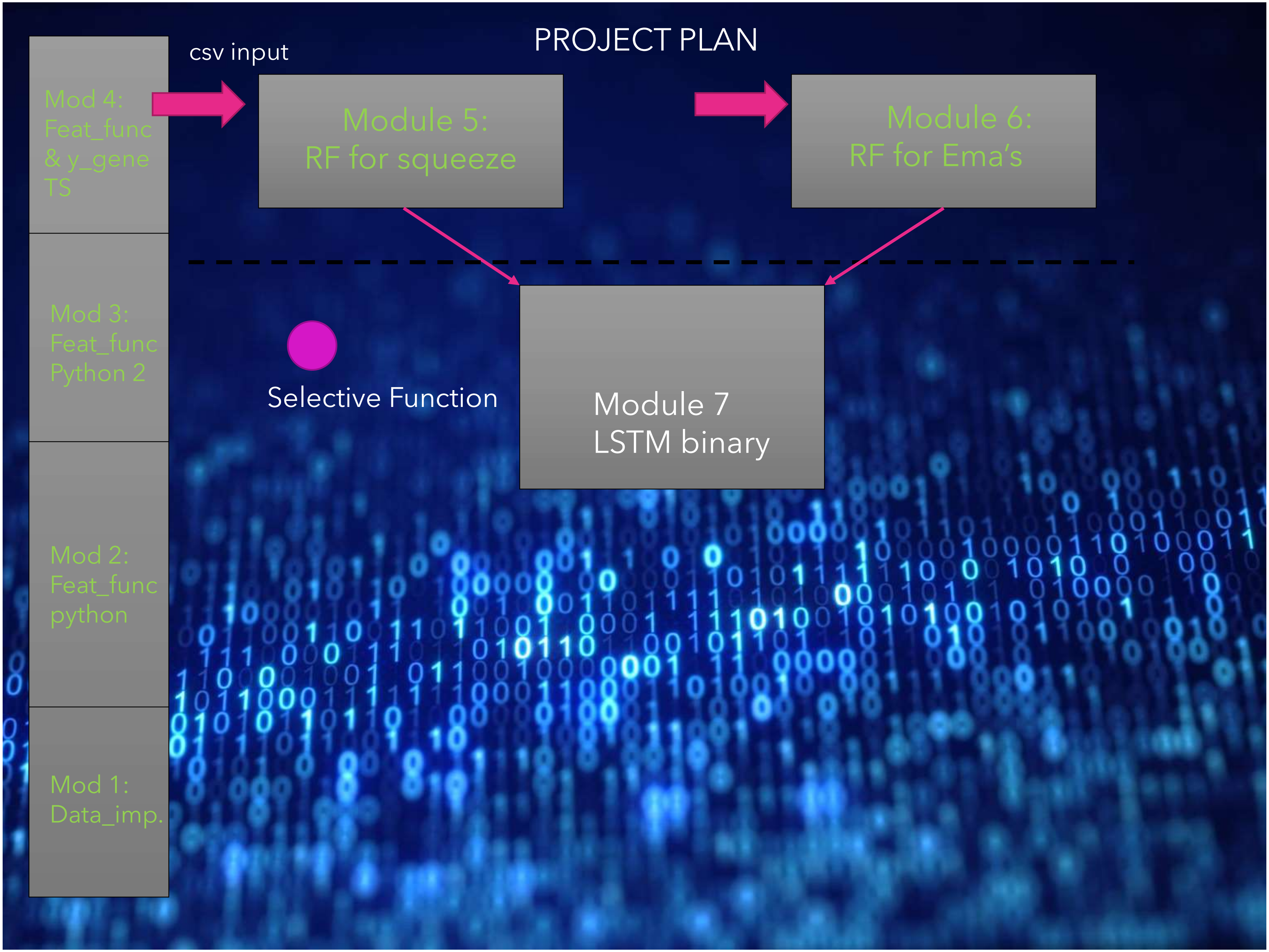
Mod 4:
Feat_func
& y_gene
TS

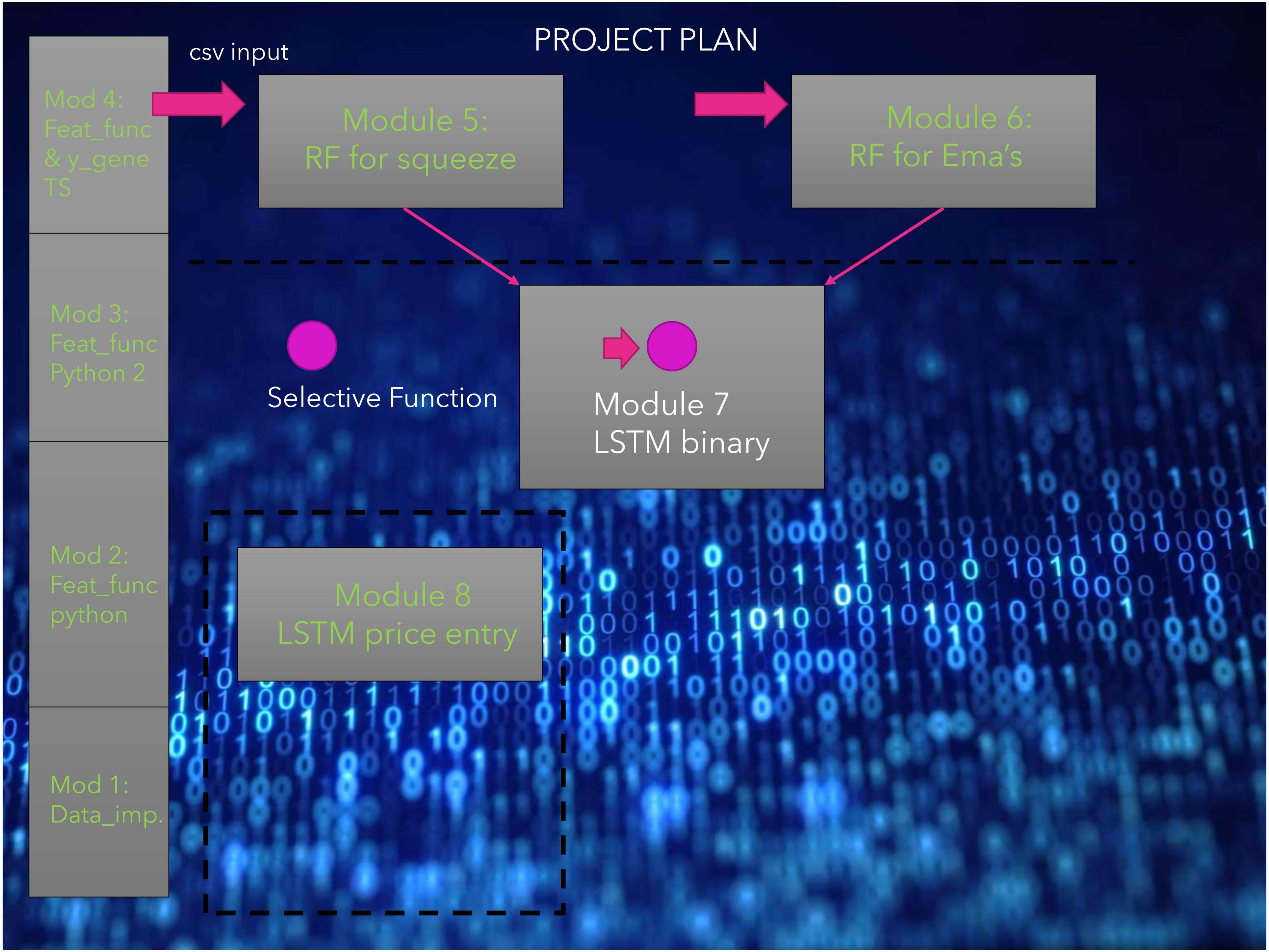
Mod 3:
Feat_func
Python 2

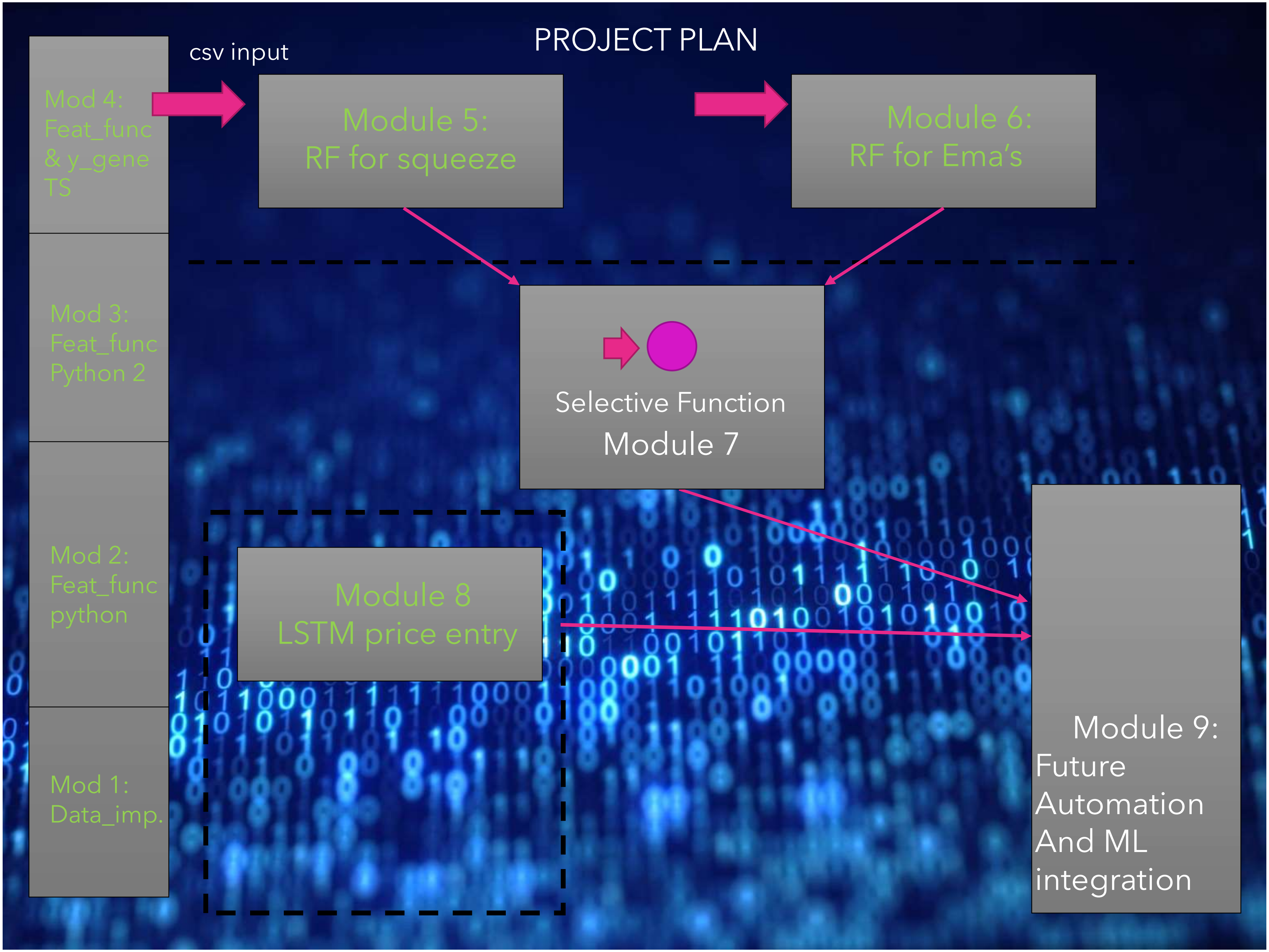
Mod 2:
Feat_func
python

Mod 1:
Data_imp.


PROJECT PLAN
csv input
Module 5:
RF for squeeze
Module 6:
RF for Ema's
Mod 4:
Feat_func
& y_gene
TS
Mod 3:
Feat_func
Python 2
Mod 2:
Feat_func
python
Mod 1:
Data_imp.


PROJECT PLAN
csv input
Mod 4: Feat_func & y_gene TS
Module 5: RF for squeeze
Module 6: RF for Ema's
Mod 3: Feat_func Python 2
Selective Function
Module 7 LSTM binary
Mod 2: Feat_func python
Mod 1: Data_imp.


PROJECT PLAN
csv input
Mod 4: Feat_func & y_gene TS
Module 5: RF for squeeze
Module 6: RF for Ema's
Mod 3: Feat_func Python 2
Selective Function
Module 7 LSTM binary
Mod 2: Feat_func python
Module 8 LSTM price entry
Mod 1: Data_imp.


PROJECT PLAN
csv input
Mod 4: Feat_func & y_gene TS
Module 5: RF for squeeze
Module 6: RF for Ema's
Mod 3: Feat_func Python 2
Selective Function
Module 7
Mod 2: Feat_func python
Module 8
LSTM price entry
Mod 1: Data_imp.
Module 9: Future Automation And ML integration

Random Forest Metrics

Module 5: Squeeze RF

Actual \ Predicted	0	1
0	TN: 1002	FP: 494
1	FN: 316	TP: 892

Precision = 64.36%

Estimators = 1000
Max_depth =25
Relevant Features 60%: bbup, bblo, kcup, kclo and 20sma.

Random Forest Metrics

Module 5: Squeeze RF

Actual \ Predicted	0	1
0	TN: 1002	FP: 494
1	FN: 316	TP: 892

Precision = 64.36%

Estimators = 1000
Max_depth =25
Relevant Features 60%: bbup, bblo, kcup, kclo and 20sma.

Module 6: EMAs cross RF

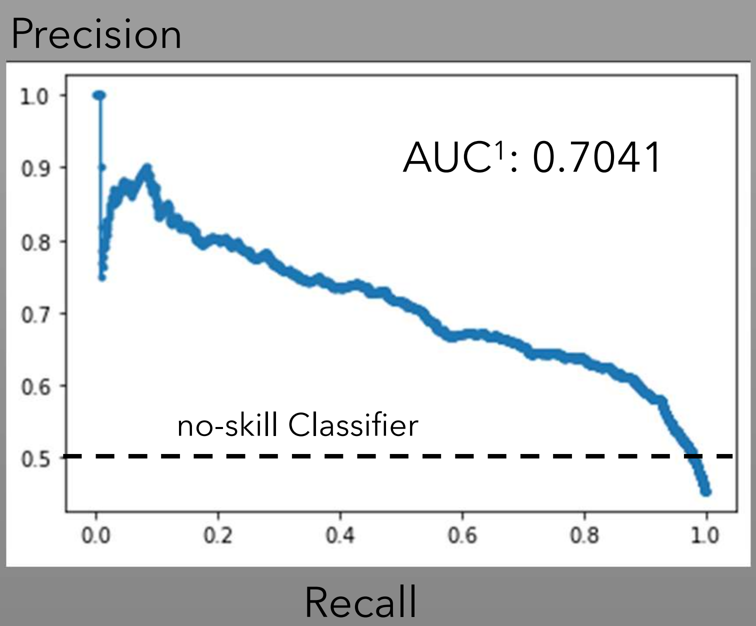
Actual \ Predicted	0	1
0	TN: 1876	FP: 758
1	FN: 20	TP: 50

Precision = 6.18%

Estimators = 1000
Max_depth =20
Relevant Features 53%: ema_dist, delta, close, ATR and high.

Random Forest Metrics

Module 5: Squeeze RF



Acc. Score: 70%

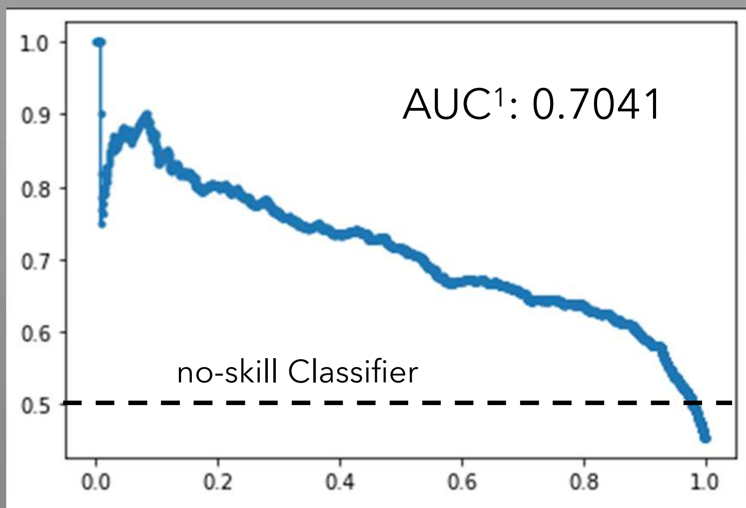
[1] AUC: Area under the curve

Precision Recall Curve

Random Forest Metrics

Module 5: Squeeze RF

Precision

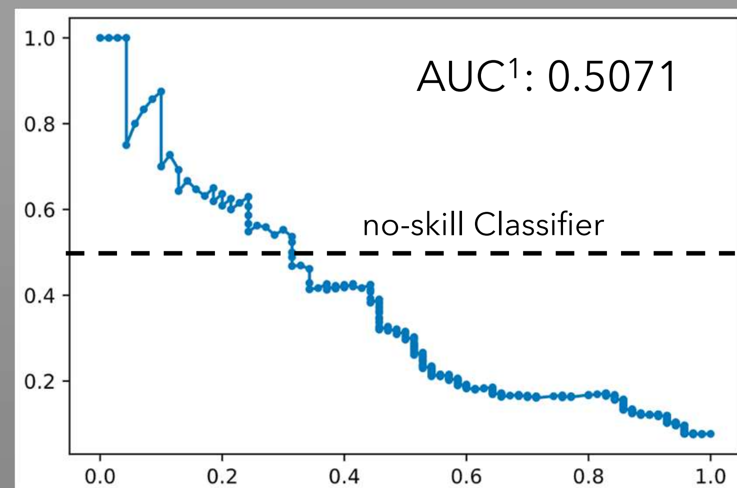


Recall

Acc. Score: 70%

Module 6: EMAs cross RF

Precision



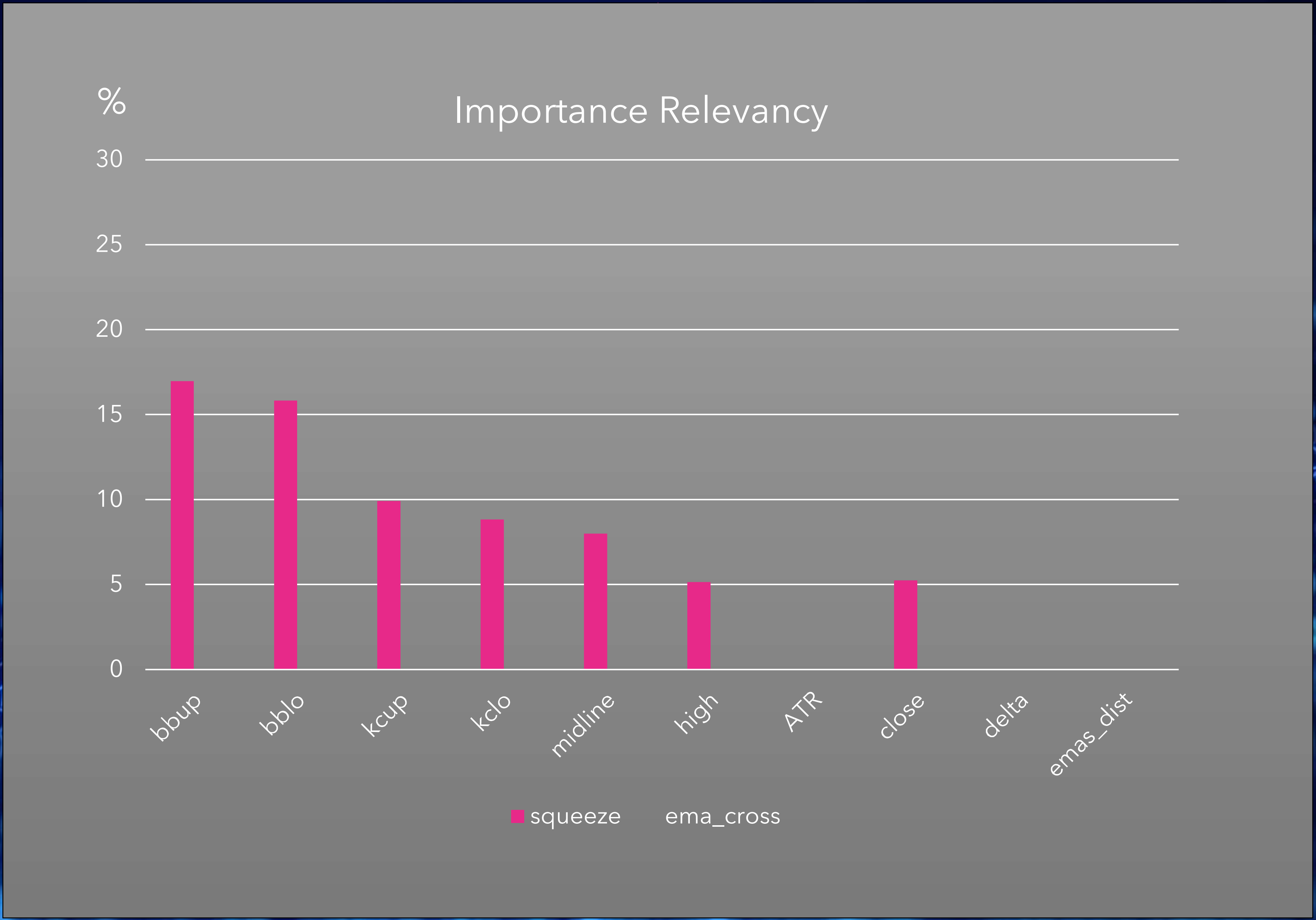
Recall

Acc. Score: 71%

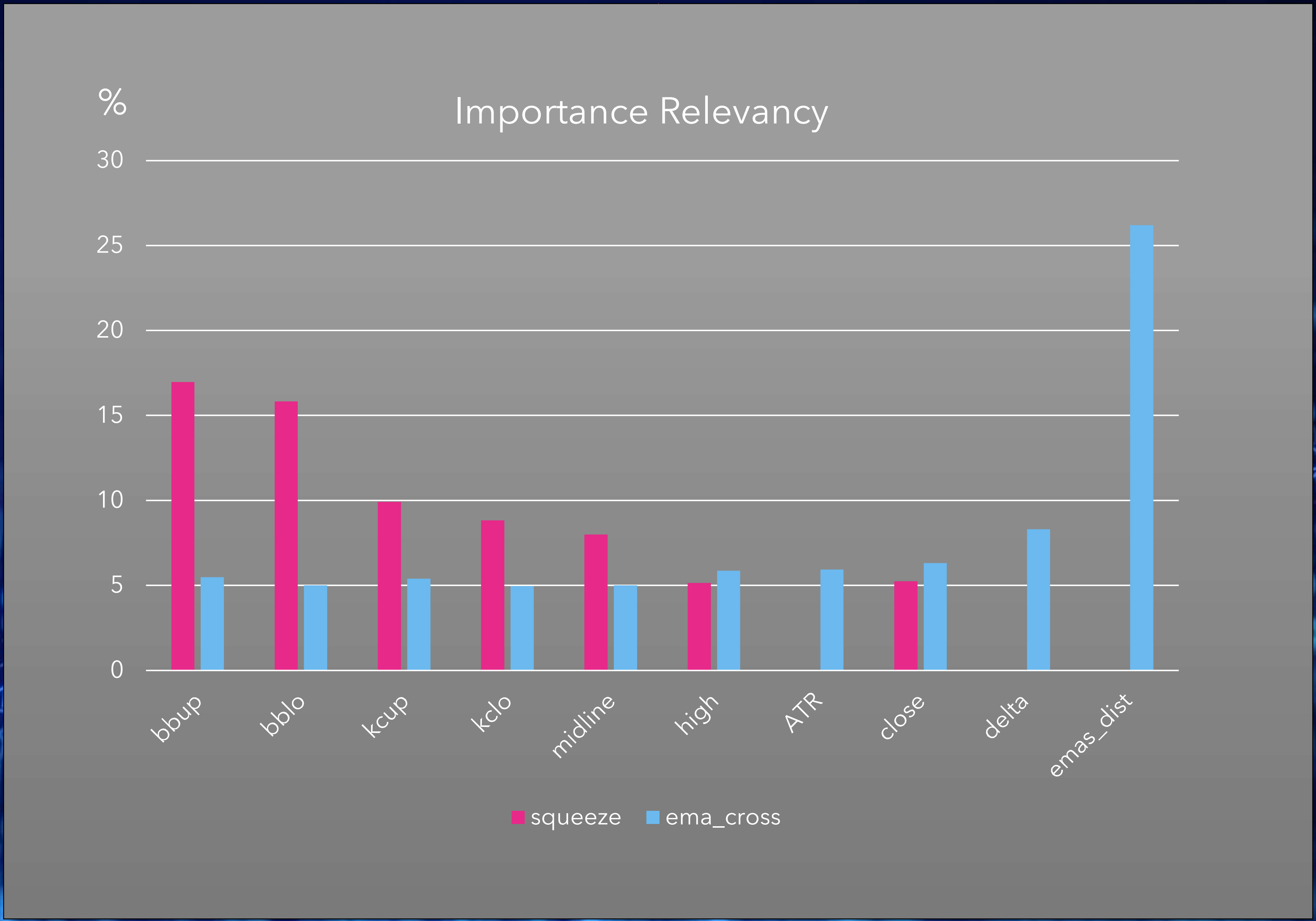
Precision Recall Curve

[1] AUC: Area under the curve

Random Forest – Importance Features



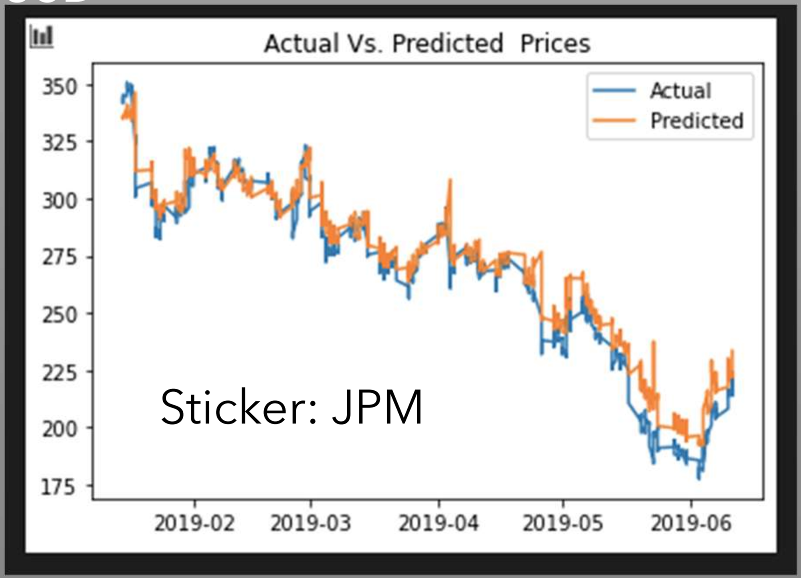
Random Forest – Importance Features



LSTM Metrics

Module 8: Price Entry

USD



Dates

Loss function = 0.00205

Future Developments

- Re-assess or reconfigure the EMA random forest to meet precision statistical requirements and improve performance.

Future Developments

- Re-assess or reconfigure the EMA random forest to meet precision statistical requirements and improve performance.
- Test models with more assets for robust statistics.

Future Developments

- Re-assess or reconfigure the EMA random forest to meet precision statistical requirements and improve performance.
- Test models with more assets for robust statistics.
- Automation / Machine Learning integration
- Develop a .py script using Select Function

Future Developments

- Re-assess or reconfigure the EMA random forest to meet precision statistical requirements and improve performance.
- Test models with more assets for robust statistics.
- Automation / Machine Learning integration
- Develop a .py script using Select Function
- Develop a Machine Learning Model to predict prices 5 or more periods ahead (target price predictor)

Future Developments

- Re-assess or reconfigure the EMA random forest to meet precision statistical requirements and improve performance.
- Test models with more assets for robust statistics.
- Automation / Machine Learning integration
- Develop a .py script using Select Function
- Develop a Machine Learning Model to predict prices 5 or more periods ahead (target price predictor)
- Develop Function that execute trades based on prices forecasted by LSTM price predictors.

Conclusions

- LSTM binary model caused barriers to the Machine Learning system.

Conclusions

- LSTM binary model caused barriers to the Machine Learning system.
- The two random forest classifiers offered 70% or above accuracy score.

Conclusions

- LSTM binary model caused barriers to the Machine Learning system.
- The two random forest classifiers offered 70% or above accuracy score.
- The EMA crossover model needs to be reconfigured to increase precision and to improve the precision-recall curve.

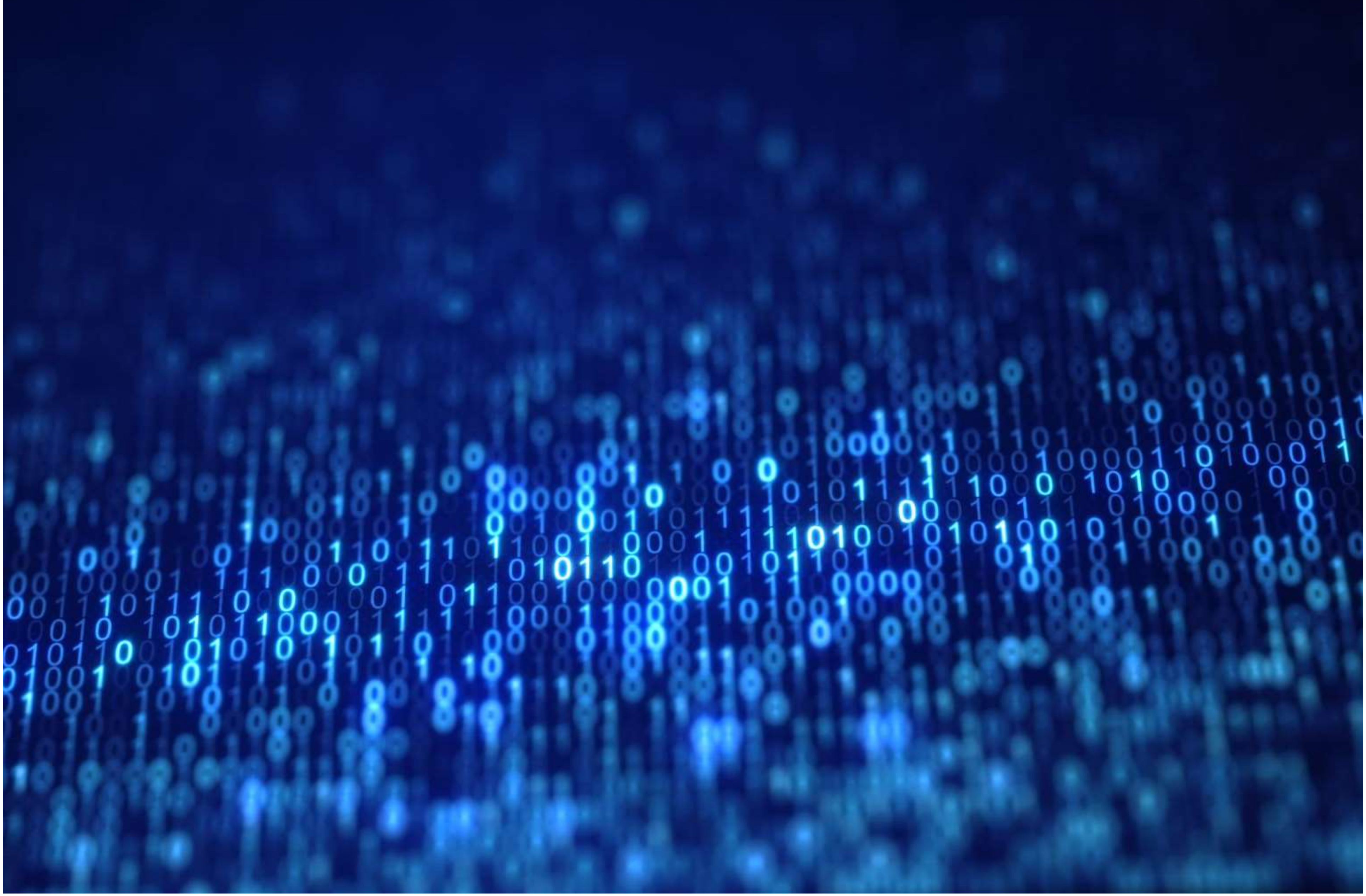
Conclusions

- LSTM binary model caused barriers to the Machine Learning system.
- The two random forest classifiers offered 70% or above accuracy score.
- The EMA crossover model needs to be reconfigured to increase precision and to improve the precision-recall curve.
- The squeeze random forest model was able to anticipate the "squeeze" state with a precision of 64%.

Conclusions

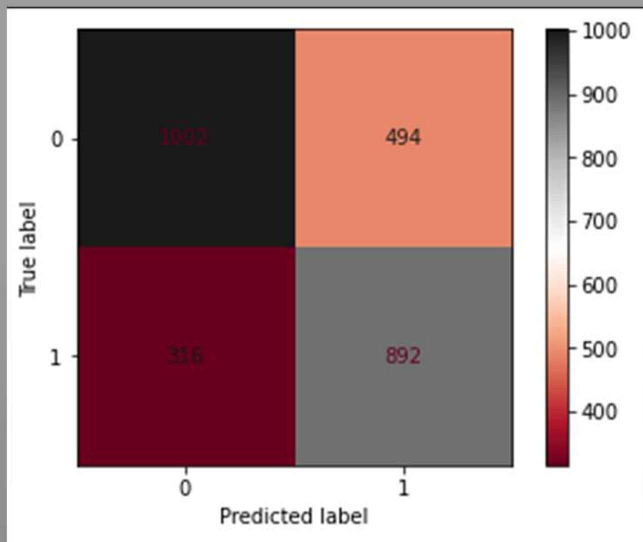
- LSTM binary model caused barriers to the Machine Learning system.
- The two random forest classifiers offered 70% or above accuracy score.
- The EMA crossover model needs to be reconfigured to increase precision and to improve the precision-recall curve.
- The squeeze random forest model was able to anticipate the “squeeze” state with a precision of 64%.
- Tests were limited to one asset and one direction (bullish trades)

algo-owls



Random Forest Metrics

Module 5: Squeeze RF



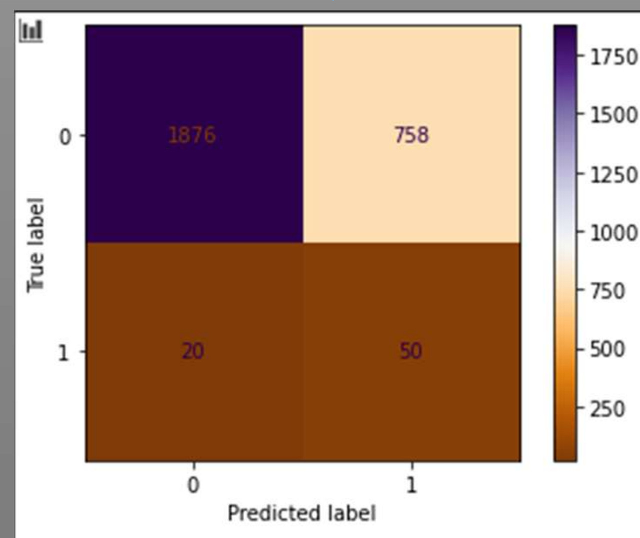
Acc. Score: 70%

Estimators = 1000
Max_depth =25
Relevant Features 60%: bbup, bblo, kcup, kclo and 20sma.

Module 6: EMAs cross RF

Estimators = 1000
Max_depth =20
Relevant Features 53%: ema_dist, delta, close, ATR and high.

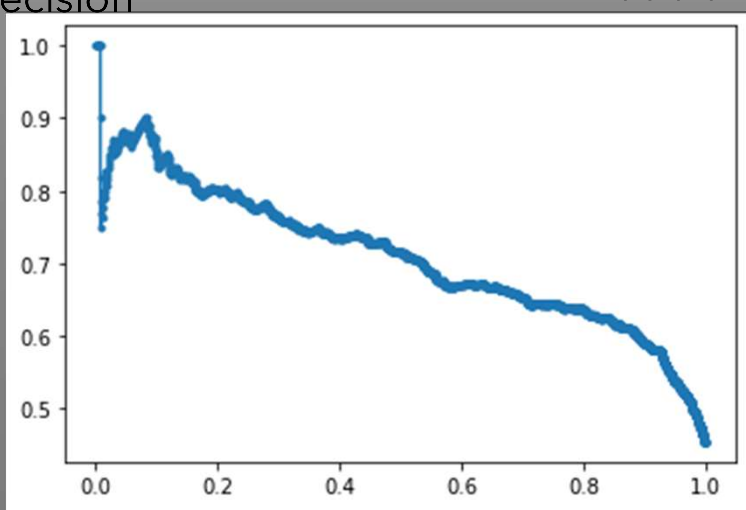
Acc. Score: 71%



Random Forest Metrics

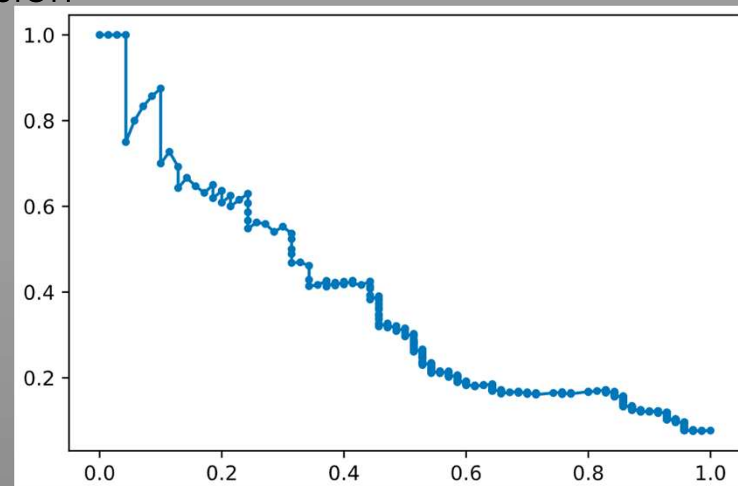
Module 5: Squeeze RF

```
Confusion Matrix

      0    1
 0  1002  494
 1   316  892
Accuracy Score 0.7004437869822485
              precision    recall  f1-score   support

           0       0.76      0.67      0.71      1496
           1       0.64      0.74      0.69      1208

    accuracy                           0.70      2704
   macro avg       0.70      0.70      0.70      2704
weighted avg       0.71      0.70      0.70      2704
```



Precision Recall Curve

Module 6: EMAs cross RF




```
Confusion Matrix

      0    1
 0  1876  758
 1    20   50
Accuracy Score 0.7122781065088757
              precision    recall  f1-score   support

           0       0.99      0.71      0.83      2634
           1       0.06      0.71      0.11        70

    accuracy                           0.71      2704
   macro avg       0.53      0.71      0.47      2704
weighted avg       0.97      0.71      0.81      2704
```

algo-owls

Confusion Matrix

We can use our confusion matrix to calculate the model's overall accuracy.

- *Accuracy* is the proportion of correct calls
- It is calculated as Acc = (TP+TN)/(TP+TN+FP+FN)
- Treats FP and FNs equally—an issue for unbalanced data

	Predicted: No (0)	**Predicted: Yes (1)**
Actual=No (0)	True Negatives (TN)	False Positive (FP)
Actual=Yes (1)	False Negative (FN)	True Positives (TP)

Confusion Matrix

We can use our confusion matrix to calculate the model's ***precision***.

- *Precision* is the proportion of positive calls that were correct.
- It is calculated as Precision = TP/(TP+FP), using the first column of the confusion matrix.
- A model with no FPs has perfect precision. All of its positive calls are correct!

If FPs are very undesirable, you want a high precision

	Predicted: No (0)	**Predicted: Yes (1)**
Actual=No (0)	True Negatives (TN)	False Positive (FP)
Actual=Yes (1)	False Negative (FN)	True Positives (TP)

Confusion Matrix

We can use our confusion matrix to calculate the model's ***recall***.

- Recall is the proportion of actually positive samples that were correct
- It is calculated as Recall = TP/(TP+FN), using the first row of the confusion matrix
- Recall is a critical metric for optimizing a model with unbalanced data
- A model with no FNs has perfect recall. All of the positive samples are correctly identified!
- Recall is sometimes called sensitivity

If FNs are very undesirable, you want a model with high recall

	Predicted: No (0)	**Predicted: Yes (1)**
Actual=No (0)	True Negatives (TN)	False Positive (FP)
Actual=Yes (1)	False Negative (FN)	True Positives (FP)

algo-owls

Confusion Matrix

We can use our confusion matrix to calculate the model's ***F1-score***.

- The *F1-score* (or F-measure) is another overall accuracy measure equivalent to the harmonic mean of the precision and recall
- It is calculated as F1 = 2* (Precision * Recall)/(Precision + Recall), using the first row and column of the confusion matrix.
- A model with perfect precision and recall has an F1-score of 1.0. The F1-score gives equal weight to precision and recall. Note that if either are 0, the score is 0 too.
- It is a good summary metric for comparing one model's performance to another.

	Predicted: No (0)	**Predicted: Yes (1)**
Actual=No (0)	True Negatives (TN)	False Positive (FP)
Actual=Yes (1)	False Negative (FN)	True Positives (TP)

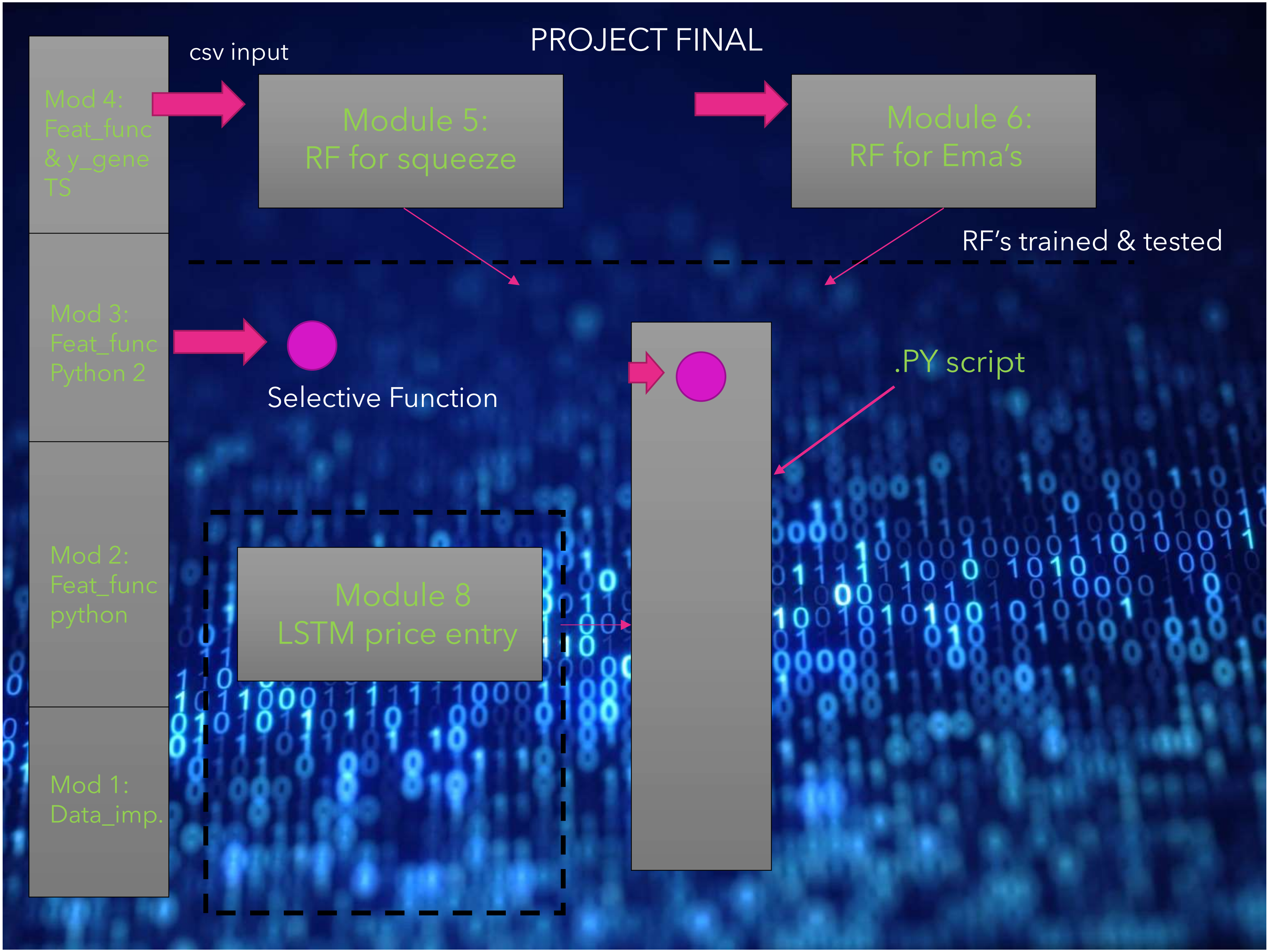
Confusion Matrix

We can use our confusion matrix to calculate the model's ***specificity***.

- *Specificity* is the proportion of actually negative samples that were correct.
- It is calculated as Specificity = TN/(TN+FP), using the second row of the confusion matrix.
- A model with no FPs has perfect specificity. All of the negative samples are correctly identified!
- Well-performing models with lots of TNs (>10,000) will often have very high specificity (>0.99).

If FPs are very undesirable, you want a highly specific model

	Predicted True	Predicted False
Actually True	TP	FN
Actually False	FP	TN


PROJECT FINAL
csv input
Mod 4:
Feat_func
& y_gene
TS
Module 5:
RF for squeeze
Module 6:
RF for Ema's
RF's trained & tested
Mod 3:
Feat_func
Python 2
Selective Function
.PY script
Mod 2:
Feat_func
python
Module 8
LSTM price entry
Mod 1:
Data_imp.