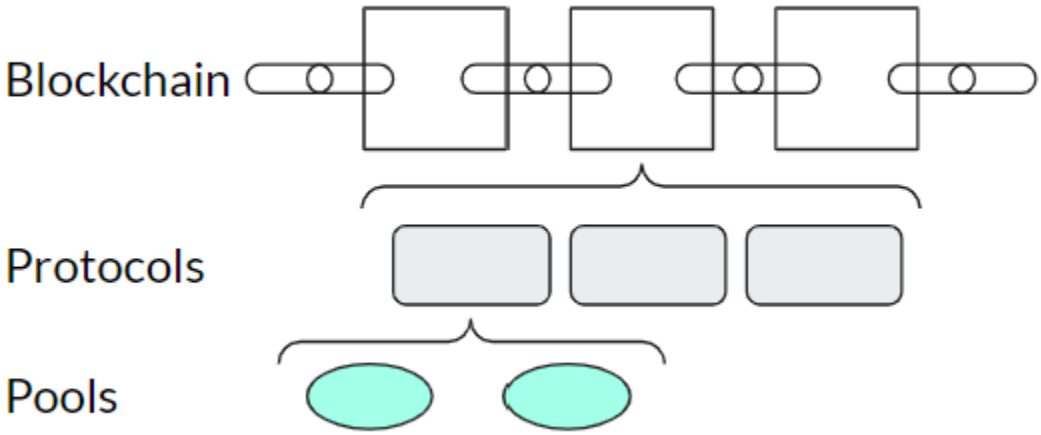
Predicting Liquidity Pool Yields using Catboost, XGBoost, and Random Forest

Introduction

The goal of this project is to use several machine-learning models to predict the fair yields on liquidity pools on public blockchains. We utilize 33 risk factors related to the pools, such as its method of generating yield, the chain and project it is built upon, historical changes in yield, and others.

Our data was gathered from DefiLlama, one of the foremost providers of decentralized finance data. Our team spoke with representatives of the company to learn more about the nature of the data.

Liquidity pools are similar to traditional finance money market accounts. A user deposits their coins into a pool in a project, or 'protocol', and the protocol's code automatically executes and generates yield in one of several ways, such as providing liquidity for decentralized exchanges, lending platforms, or staking native coins to secure the blockchain it is built on. Liquidity pools are a catch-all term for these applications within the decentralized finance ecosystem.



Labels and Features

The label used in the model is:
Yield APY (apy) - float - The annual percentage yield of the pool

The complete list of the model's features are:

- Pool Total Value Locked USD (tvlUsd) - float - the dollar amount of capital staked in the pool
- Reward Tokens (rewardTokens) - dummy - whether or not receiving additional native tokens of the protocol as part of the yield is part of the protocol or not
- Change in APY Over Previous One Day (apyPct1D) - float - the percent change of the APY over the previous day
- Change in APY Over Previous Seven Days (apyPct7D) - float - the percent change of the APY over the previous seven days
- Change in APY Over Previous ThirtyDays (apyPct30D) - float - the percent change of the APY over the previous thirty days
- Cryptocurrency in Pool is a Stablecoin (stablecoin) - dummy - whether or not the cryptocurrency in the pool is a stablecoin
- Impermanent Loss (ilRisk) - dummy - whether or not the pool experience impermanent loss, which is the value lose in the pool by arbitrageurs keeping prices and ratios stable
- Pool Utilizes One or Multiple Coins (single_coin_exposure) - dummy - whether or not the pool uses one coin or more than one coin
- DefiLlama Prediction of Dropping or Stable/Rising APY(apy_predicted_state) - polychotomous dummy - whether the prediction of APY is Dropping (-1), No data available (0) or Stable/Rising (1)
- DefiLlama Probability of APY Prediction Happening (apy_predicted_state_prob) - polychotomous dummy - the probability DefiLlama publishes of their prediction coming true
- DefiLlama Algorithm Prediction of Stability (binnedConfidence) - polychotomous dummy - to what extent DefiLlama's calculated probability is stable (0-3)
- Total APY Average over Previous 7 Days (mu) - float - the average of total APY over last seven days
- Total APY Volatility over Previous 7 Days (sigma) - float - the volatility of daily total APY over last seven days
- Number of Pools in Project/Protocol (count) - int- the total number of pools within a project
- Outlier Pools (outlier) - dummy - pools that DefiLlama considers as outliers bases on geometric mean of APY values
- Impermanet Loss over Previous 7 Days (il7d) - float - the percentage loss for liquidity providing over the last 7 days vs. holding the underlying assets instead (only relevant for decentralized exchanges)
- DEx APY over Previous 7 Days (apyBase7d) - float - APY based on trading fees over the past 7 days (only relevant for decentralized exchanges)
- Total APY Average over Previous 30 Days (apyMean30d) - float - the average of total APY over last thirty days
- Total Value Locked of Blockchain (chain_tvl) - float - the dollar amount of all of the capital locked in Defi protocols on that certain chain
- Total Value Locked of Project/Protocol (project_tvl) - float - the dollar amount of all of the capital locked in pools within a certain project/protocol

- Auditied Project - (project_audited) - dummy - whether or not the protocol has been audited by a reputable auditing company verified by DeFiLlama
- Decentralized Exchange (dex) - dummy - whether or not the pool generates its yield from being a DEx
- Lending (lending) - dummy- whether or not the pool generates its yield from being a lending protocol
- Yeld (yield) - dummy - whether or not the pool generates its yield from being a yield-boosting protocol built on top of another protocol
- Yield Aggregators (yield_agg) - dummy - whether or not the pool generates its yield from being part of a yield aggregator, or a protocol made up of several sub-protocols that work in tandem with eachother
- Collateralized Debt Position (cdp) - dummy- whether or not the pool generates its yield from offering users CDPs to stake their coins and receive stablecoins back
- Liquid Staking (liquid_staking) - dummy - whether or not the pool generates its yield from offering users a service to stake their assets and receive yield and liquidity provider (LP) tokens. This staking is toward the security of the actual blockchain, not a single protocol or pool.
- Multichain - (project_multichain) - dummy - whether or not the project/protocol the pool is within is present on a single chain or multiple chains
- Change in Project Total Value Locked over One Hour (project_tvl_change_1h) - float - the percentage change of the TVL of a project over the past hour
- Change in Project Total Value Locked over One Day (project_tvl_change_1d) - float - the percentage change of the TVL of a project over the past day
- Change in Project Total Value Locked over One Week (project_tvl_change_7d) - float - the percentage change of the TVL of a project over the past 7 days

Models Used

Catboost, XGBoost, and Random Forest
CatBoost is a machine learning library open sourced by the Russian search giant Yandex in 2017, and it is a type of Boosting family algorithm. CatBoost, XGBoost, and LightGBM are also known as the three mainstream artifacts of GBDT, all of which are improved implementations under the framework of the GBDT algorithm. XGBoost is widely used in the industry. LightGBM effectively improves the computational efficiency of GBDT. Yandex's CatBoost claims to be a better algorithm than XGBoost and LightGBM in terms of algorithm accuracy.

CatBoost is a GBDT framework based on symmetric decision trees (oblivious trees) as a base learner with fewer parameters, support for categorical variables and high accuracy. As can be seen from its name, CatBoost is composed of Categorical and Boosting. In addition, CatBoost also solves the problems of gradient bias (Gradient Bias) and prediction shift (Prediction shift), thereby reducing the occurrence of overfitting, thereby improving the accuracy and generalization ability of the algorithm.

Compared with XGBoost and LightGBM, the innovations of CatBoost are:

An innovative algorithm that automatically converts categorical features into numerical features is embedded. First, do some statistics on categorical features, calculate the frequency of a certain category feature (category), and then add hyperparameters to generate new numerical features (numerical features).
Catboost also uses combined category features, which can take advantage of the connection between features, which greatly enriches the feature dimension.
The method of ranking promotion is used to fight against the noise points in the training set, so as to avoid the deviation of gradient estimation, and then solve the problem of prediction deviation.
A fully symmetric tree is used as the base model.

There is a biggest advantage in Catboost. With high-cardinality features such as user ID, the one-hot encoding will have dimension disaster. In the Catboost, we use target statistics to deal with this problem. The mechanism is mapping the target value y to the label X. The easiest way is to assign the average value to the groups with different labels.

Model Evaluation

Catboost

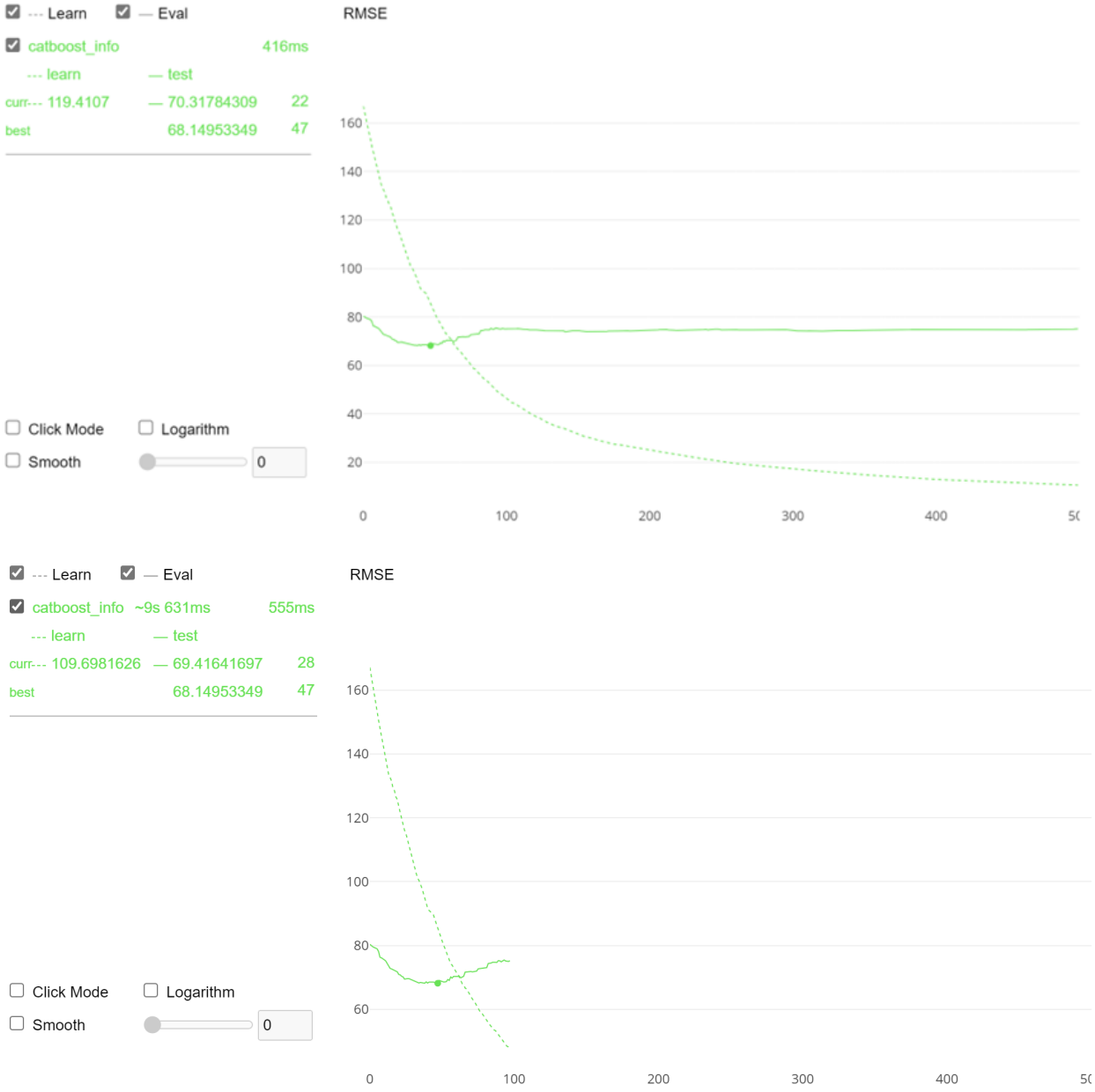
Sheet

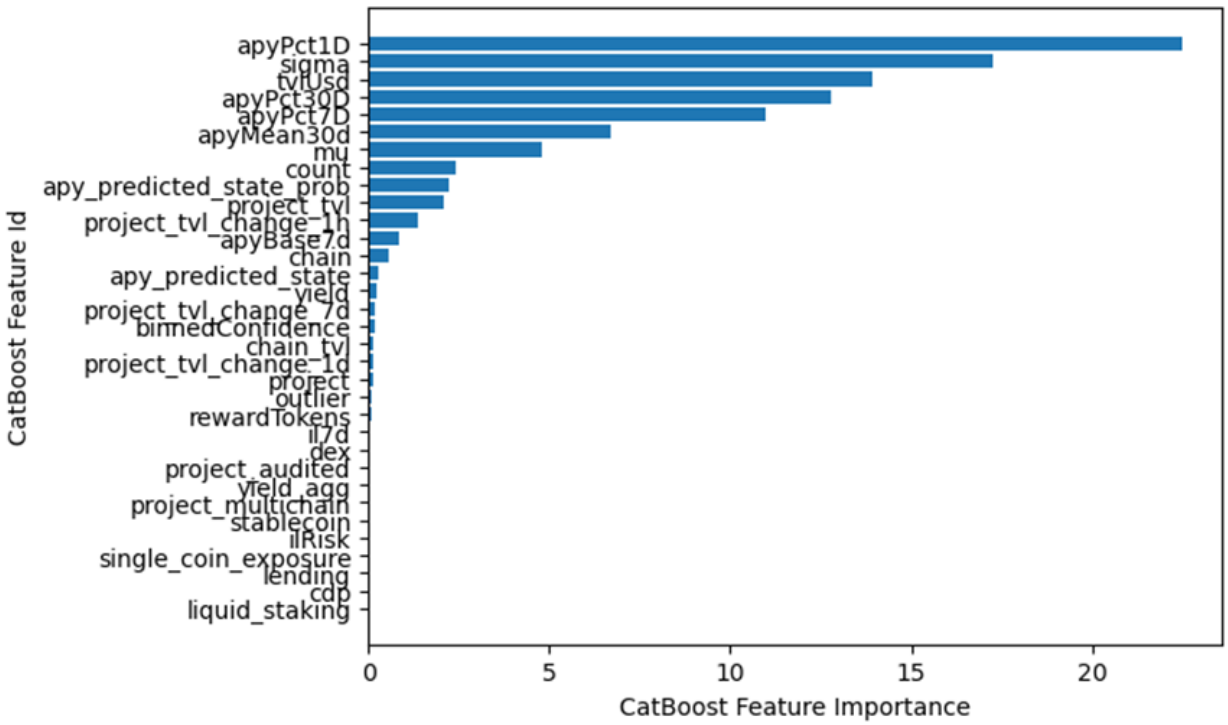
Grid Search	**range**	**Best Parameters**
Depth	[4, 6, 8]	4
Iterations	[200, 500, 1000]	500
Learning_rate	[0.03, 0.01]	0.03
L2_leaf_reg	[0.5, 1, 3]	0.5

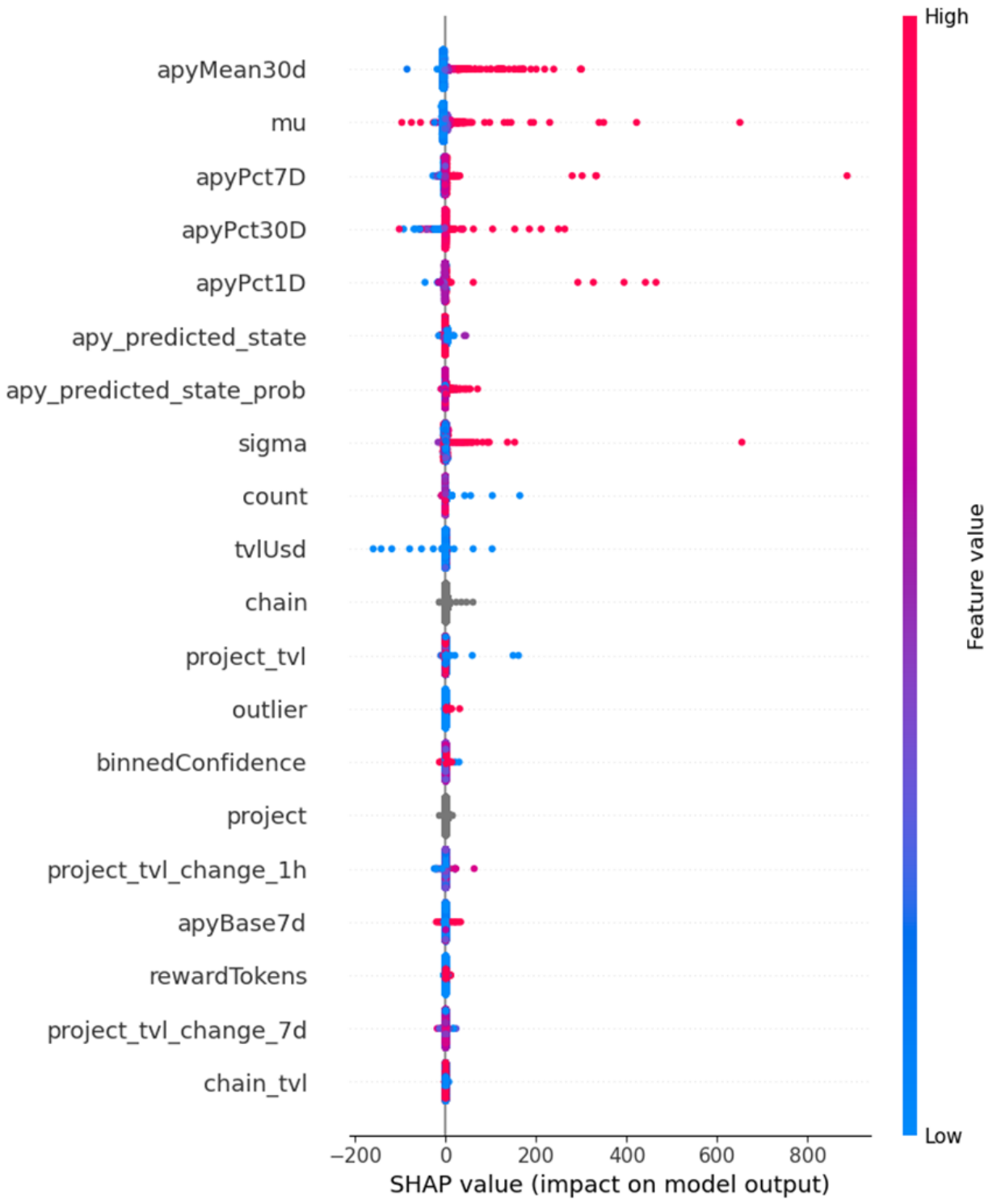
Depth: Depth of the tree.
Iterations: The maximum number of trees that can be built when solving machine learning problems.
learning_rate: Used for reducing the gradient step.

l2_leaf_reg: Coefficient at the L2 regularization term of the cost function. Any positive value is allowed.

Based on the model with the best parameters, our testing performance is RMSE: 67.96 and R2: 0.56.
The picture below is how the model is fitted in the train set and optimized in the validation set.




apyPct1D
sigma
tvlUsd
apyPct30D
apyPct7D
apyMean30d
mu
count
apy_predicted_state_prob
project_tvl
project_tvl_change_1h
apyBase7d
chain
apy_predicted_state
yield
project_tvl_change_7d
binnedConfidence
chain_tvl
project_tvl_change_1d
project
outlier
rewardTokens
il7d
dex
project_audited
yield_agg
project_multichain
stablecoin
ilRisk
single_coin_exposure
lending
cdp
liquid_staking
CatBoost Feature Id
0
5
10
15
20
CatBoost Feature Importance



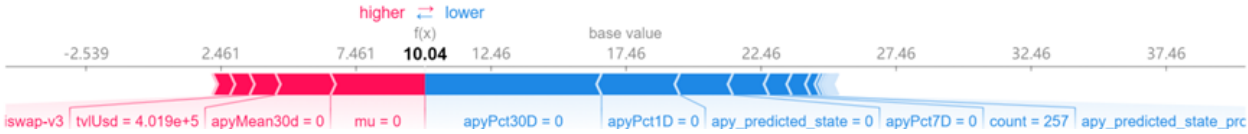
In the SHAP value picture, we can see when the features have more red points on the right of 0 y-axis, it means the feature have positive correlation with the model output.

combining the CatBoost Feature Importance picture with SHAP value picture above, we can see features apyPect1D, apyPect30D and apyPect7D all have big and positive contributions to apy.

Which shows the momentum in these features. It is worthwhile to mention that Sigma has a risk premium to apy, which means higher risk(volatility), higher return.

Let's have a deeper look at how the individual SHAP values are constructed. There are two examples below.

This one shows the sigma value and BSC chain have a positive influence on the output. Other blue ones have negative influences although they are mostly equal to 0 which are the median of most features.



In this example, except the ones with value 0, we can see tvlUsd(the total value locked in USD) have positive contribution on the output and also the project(protocol) uniswap-v3.

Comparison

	Catboost	Xgboost	RandomForest
Training R2	0.994992551736271	0.9984887342544649	0.8427103716587185
Test R2	0.5575001629868552	0.6031760912179874	0.5010015648502459

Applications

While this project may be considered a proof-of-concept, the method of generating alpha with these models is quite straightforward. When determining where to efficiently allocate capital within the decentralized finance ecosystem to maximize the risk-adjusted returns, we utilize our model to predict fair yields based on current values of risk factors. If the actual yield exceeds our predicted fair yield, we interpret this as the market overestimating the pool risk and thus setting the yield APY too high relative to its lower risk. So, we would allocate capital into this

pool and generate yield with the higher APY before the actual yield converges to our predicted fair yield.

Vice versa, if the actual APY is lower than our predicted APY, we assume the market underestimates the risk of the pool and therefore sets the APY too low. In this situation, we would not allocate capital since the APY is too low relative to the risk of the pool.