Cryptocurrency

Price Prediction Machine

Learning Project Report

1. DATA TRANSFORMATION & FEATURE ENGINEERING

1.1 Original Dataset

The project began with a raw cryptocurrency dataset containing 111,096 records with price, volume, and change data across multiple cryptocurrencies over 7 months (March to October 2025).

1.2 Data Preprocessing Steps

Here is the content converted into clear bullet points:

- **Step 1-3: Numeric Conversion**
  - Converted string-formatted numbers (e.g., with $, %, ,) to proper numeric types.
  - Handled unit suffixes (K, M, B, T) by converting them to actual numeric values.

- **Step 4: Missing Value Imputation**
  - Applied forward and backward fill grouped by cryptocurrency to maintain coin-specific patterns.
  - Total missing values handled: **91** (0.08%).

- **Step 5: Duplicate Removal**
  - Removed **87** duplicate timestamp-symbol combinations.

- **Step 6: Invalid Value Handling**
  - Removed **522** rows with zero or negative prices or zero market cap.

- **Step 7: Feature Engineering – Volume-to-Market-Cap Ratio**
  - Created new feature: vol_to_marketcap_ratio = vol_24h / market_cap.
  - Indicates trading activity relative to market size.

- **Step 8: Binary Target Creation**
  - Created target variable:
    - 1 if chg_24h > 0 (price increased),
    - 0 if chg_24h ≤ 0 (price decreased).

- **Step 9: Future Target Creation (Critical!)**
  - Created future_target by shifting target forward to prevent data leakage (see Section 3 for details).

1.3 New Features Introduced

Feature	Type	Description	Purpose
vol_to_marketcap_ratio	Continuous	Volume / Market Cap	Trading activity indicator
target	Binary (0/1)	Current period movement	Original target (has leakage)
future_target	Binary (0/1)	NEXT period movement	Correct target (no leakage)

1.4 Final Dataset

The final dataset contain following statistics

- **Rows**: 110,189
- **Columns**: 12
- **Data Retention**: **99.18%** of original dataset
- These are the following features

Feature Name	Description
`timestamp`	Date and time of the data record
`name`	Name of the asset (e.g., Bitcoin, Ethereum)
`symbol`	Asset ticker symbol (e.g., BTC, ETH)
`price_usd`	Current price in USD
`vol_24h`	24-hour trading volume
`total_vol`	Total trading volume (definition may vary; verify distinction from `vol_24h`)
`chg_24h`	24-hour price change (could be percentage or absolute value)
`chg_7d`	7-day price change
`market_cap`	Market capitalization
`vol_to_marketcap_ratio`	Ratio of trading volume to market cap — an indicator of liquidity
`target`	Target variable (likely a current state or classification label)
`future_target`	Future state or label used for predictive modeling

2. INITIAL ISSUE: DATA LEAKAGE DETECTED

2.1 The Problem

Initial model training achieved 100% accuracy, which raised a red flag. Upon investigation, I discovered data leakage - the model was using information that would not be available at prediction time.

2.2 Root Cause Analysis

The target variable was created as: target = 1 if chg_24h > 0, else 0. However, I was using chg_24h as a feature to predict this target. This created a circular relationship:

- **Feature:** chg_24h (today's price change)
- **Target:** target (1 if chg_24h > 0)
- **Problem:** The model was learning: 'if chg_24h > 0, predict 1' (trivial!)

2.3 Evidence of Data Leakage

Indicator	Value	Assessment
Training Accuracy	100.00%	Suspiciously perfect
Test Accuracy	100.00%	No overfitting gap
Feature Importance (chg_24h)	96.15%	Extreme dominance
Simple Rule Accuracy	100.00%	Same as a complex model!

A simple rule 'if chg_24h > 0, predict UP' achieved the same 100% accuracy, proving the model was not learning complex patterns but merely the target definition.

2.4 Impact Assessment

- Model was not production-ready (predicting past, not future)
- 100% accuracy was meaningless (circular logic)
- Would fail completely in real-world deployment
- Required fundamental redesign of target variable

3. SOLUTION: FIXING DATA LEAKAGE

3.1 The Fix: Future Target Creation

The solution involved creating a new target variable that represents FUTURE movement rather than current movement. This transforms the problem from 'describing the past' to 'predicting the future'.

3.2 Implementation: Shift Operation

```
df['future_target'] = df.groupby('symbol')['target'].shift(-1)
```

The key transformation was implemented using pandas shift operation grouped by cryptocurrency:

3.3 How the Shift Works

- **Step 1: Group by Symbol**
  Organize the data by each cryptocurrency (e.g., BTC, ETH) to keep their time series separate.

- **Step 2: Shift Target Forward**
  Within each group, shift the target column one row up (`shift(-1)`) so that the target represents the next time period's value.

- **Step 3: Remove Last Rows**
  Drop the last row of each group, since it now has a missing (`NaN`) target due to the shift.

- **Step 4: Final Result**
  Now, each row contains today's features paired with the target for the next time period—ready for predictive modeling.

3.4 Example Transformation

Time	Symbol	chg_24h	OLD target	NEW future_target	Meaning
10:00	BTC	+2.19%	1 (up)	1	U e 10:00 data ⟶ predict 11:00
11:00	BTC	+2.16%	1 (up)	0	U e 11:00 data ⟶ predict 12:00
12:00	BTC	+2.12%	1 (up)	Removed	Cannot predict beyond data

3.5 Why Grouping by Cryptocurrency is Critical

Without groupby('symbol'), the last row of Bitcoin would predict the first row of Ethereum (wrong!). Grouping ensures each cryptocurrency's future is predicted using only its own data, maintaining temporal and logical consistency.

3.6 Data Changes Summary

Aspect	Before	After
Rows	110,487	110,189 (-298)
Target Variable	target (current)	future_target (next)
Prediction Type	Describes past	Predicts future
Data Leakage	Yes	No

4. MODEL SELECTION & JUSTIFICATION

4.1 Chosen Model: Random Forest Classifier

1. **Effective for Complex, Volatile Data**: Random Forest handles non-linear relationships and is robust to outliers—ideal for the volatile and unpredictable nature of cryptocurrency markets.

2. **Low Preprocessing & Interpretability**: It requires no feature scaling and offers built-in feature importance, making the model both simpler to prepare and more interpretable.

3. **Efficient & Reliable**: The ensemble approach reduces overfitting, enables fast training via parallel processing, and has a strong track record in financial prediction tasks.

5. MODEL PERFORMANCE ON TEST DATA

5.1 Training Configuration

- **Dataset:** refined_data.csv (110,189 rows)
- **Features:** chg_24h, chg_7d, vol_to_marketcap_ratio
- **Target:** future_target (binary: 0=down, 1=up)
- **Split:** 80% training (88,151 samples), 20% testing (22,038 samples)
- **Stratified:** Yes (maintains class balance in both sets)

5.2 Overall Performance Metrics

Metric	Value	Interpretation
Test Accuracy	88.95%	Excellent - 89 out of 100 predictions correct
Precision	89.54%	When predicting UP, correct 90% of time
Recall	87.95%	Catches 88% of all coins that actually go up
F1-Score	0.89	Excellent balance between precision and recall

5.2 Confusion Matrix

	Predicted DOWN	Predicted UP	Total
Actual DOWN	10,002	1,121	11,123
Actual UP	1,315	9,600	10,915
Total	11,317	10,721	22,038

Analysis: True Positives (9,600) + True Negatives (10,002) = 19,602 correct predictions out of 22,038 = 88.95% accuracy. False Positives (1,121) and False Negatives (1,315) represent the 11% error rate.

5.3 Class-wise Performance

Class	Precision	Recall	F1-Score	Support
DOWN (0)	88%	90%	0.89	11,123
UP (1)	90%	88%	0.89	10,915

Both classes show balanced performance with ~89% precision, recall, and F1-score, indicating no bias toward either class.

5.4 Feature Importance

Rank	Feature	Importance	Interpretation
1	chg_24h	91.32%	Today's momentum is strongest predictor
2	chg_7d	5.98%	Weekly trend provides context
3	vol_to_marketcap_ratio	2.71%	Trading activity has minor impact