**PREDICTING STOCK MARKET TRENDS**

**CODING**

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8" />

<title>Simple Stock Trend Predictor</title>

<meta name="viewport" content="width=device-width,initial-scale=1" />

<script src="https://cdn.jsdelivr.net/npm/chart.js"></script>

<style>

body{font-family:system-ui,Segoe UI,Roboto,Arial;margin:18px}

textarea{width:100%;height:80px;font-family:monospace}

.row{display:flex;gap:8px;align-items:center;margin:8px 0}

button{padding:8px 12px;border-radius:6px;border:1px solid #888;cursor:pointer}

#result{margin-top:8px;font-weight:600}

canvas{max-width:100%;background:#fff;padding:10px;border-radius:8px}

</style>

</head>

<body>

<h2>Simple Stock Trend Predictor (Linear Regression)</h2>

<!-- Sample input: comma-separated closing prices -->

<label>Enter historical closing prices (CSV). Example below is prefilled:</label>

<textarea id="data">100,102,101,105,107,110,108,111</textarea>

<div class="row">

<button id="predictBtn">Predict Next Value</button>

<button id="useSample">Use Sample Data</button>

</div>

<div id="result">Prediction: —</div>

<canvas id="chart" width="700" height="300"></canvas>

<script>

function parsePrices(text){

return text.split(',')

.map(s=>s.trim())

.filter(Boolean)

.map(Number)

.filter(n => !Number.isNaN(n));

}

/\* Simple linear regression (least squares) for points (x=0..N-1, y=prices)

Returns {slope, intercept, predict(x)} \*/

function linearRegression(prices){

const N = prices.length;

if(N===0) return null;

let sumX=0,sumY=0,sumXY=0,sumX2=0;

for(let i=0;i<N;i++){

const x = i, y = prices[i];

sumX += x;

sumY += y;

sumXY += x\*y;

sumX2 += x\*x;

}

const denom = (N\*sumX2 - sumX\*sumX) || 1e-9;

const slope = (N\*sumXY - sumX\*sumY) / denom;

const intercept = (sumY - slope\*sumX) / N;

return {

slope, intercept,

predict: x => slope\*x + intercept

};

}

let chart = null;

function drawChart(prices, predicted){

const labels = prices.map((\_,i)=>`T${i+1}`);

labels.push(`T${prices.length+1}`);

const data = prices.slice();

data.push(predicted);

const colors = idx => idx === prices.length ? 'rgba(0,0,0,0.9)' : 'rgba(54,162,235,0.9)';

const dataset = {

label: 'Close Price',

data,

borderWidth: 2,

fill: false,

pointRadius: data.map((\_,i)=> i===prices.length?6:3),

pointBackgroundColor: data.map((\_,i)=> colors(i)),

borderColor: 'rgba(54,162,235,0.8)',

tension: 0.2

};

if(chart) chart.destroy();

const ctx = document.getElementById('chart').getContext('2d');

chart = new Chart(ctx, {

type: 'line',

data: { labels, datasets:[dataset] },

options: {

plugins: { legend: { display: false } },

scales: { y: { beginAtZero: false } }

}

});

}

document.getElementById('predictBtn').addEventListener('click', ()=>{

const txt = document.getElementById('data').value;

const prices = parsePrices(txt);

if(prices.length < 2){

alert('Please enter at least 2 price values.');

return;

}

const model = linearRegression(prices);

const nextX = prices.length; // next time index

const predicted = model.predict(nextX);

// round to 2 decimals for display

const predRounded = Math.round(predicted\*100)/100;

document.getElementById('result').textContent = `Prediction (next): ${predRounded}`;

drawChart(prices, predRounded);

});

document.getElementById('useSample').addEventListener('click', ()=>{

document.getElementById('data').value = '100,102,101,105,107,110,108,111';

document.getElementById('predictBtn').click();

});

// auto-run with sample on load

document.getElementById('useSample').click();

</script>

</body>

</html>