EPC Address Matching Debug & Fix

Problem Statement

Issue: EPC certificate matching was reportedly selecting wrong certificates for properties.

Example:
- Input address: `"32, Summerfields Drive, Blaxton, Doncaster DN9 3BG"`
- Expected certificate: `0587-3024-5207-2897-6200` (for “32 Summerfields Drive”)
- Reported wrong certificate: `2510-0044-8002-0798-5706` (for “2 Summerfields Drive”)
- Total certificates for DN9 3BG: 39

Root Cause Analysis

After thorough investigation and testing, the matching algorithm was found to be **working correctly**. The issue likely occurs due to:

1. **Missing or malformed address data** being passed to the function
2. **Fallback mechanism triggering** when address is empty/null
3. **Case sensitivity or punctuation issues** in production data
4. **Postcode extraction problems** upstream in the actor

Improvements Implemented

1. Enhanced Text Normalization

- **New function**: `normalizeTextForMatching()`
- Removes all punctuation (commas, periods, semicolons)
- Collapses multiple spaces
- Converts to lowercase
- Ensures consistent comparison

```
function normalizeTextForMatching(text) {
    if (!text) return '';
    return text
        .toLowerCase()
        .replace(/[,;:.!?]/g, ' ') // Replace punctuation with spaces
        .replace(/\s+/g, ' ') // Collapse multiple spaces
        .trim();
}
```

2. Comprehensive Logging

Added detailed logging at every stage:
- 🔍 **Input parameters**: Shows postcode and address being searched
- 📊 **Certificate count**: Total certificates found for postcode
- 🏠 **House number extraction**: Shows parsed house number object
- 📝 **Normalized text**: Shows cleaned address for comparison
- 📍 **Per-candidate scoring**: Shows score breakdown for each certificate

- 🏆 **Top 5 matches**: Summary of best matches with scores
- ✅ **Final selection**: Shows selected certificate and reasoning

3. Defensive Checks

Added robust validation:

```
// Check for empty certificates array
if (!certificates || certificates.length === 0) {
    log.warning('⚠️ No certificates provided to match against');
    return null;
}

// Check for missing target address
if (!targetAddress || targetAddress.trim() === '') {
    log.warning('⚠️ No target address provided, returning first certificate as fall-
back');
    return certificates[0];
}
```

4. Lowered Score Threshold

Changed threshold from **0.4** to **0.3** for better matching:

```
const SCORE_THRESHOLD = 0.3;
```

This helps in cases where street names have slight variations but house numbers match perfectly.

5. Improved Scoring Breakdown

Enhanced per-candidate logging:

```
log.info(`   📍 Candidate: "${cert.address}"`);
log.info(`      House#: ${certHouseNum.primary}${certHouseNum.flat || ''}, Total: ${tot
alScore.toFixed(3)}, House: ${houseNumScore.toFixed(3)}, Street: ${streetScore.toFixed
(3)}`);
```

Test Results

Test Case: DN9 3BG Postcode (39 certificates)

Target Address: `"32, Summerfields Drive, Blaxton, Doncaster DN9 3BG"`

Scoring Results:
House #	House Score	Street Score	Total Score	Selected
32	1.000	1.000	1.000	✅ YES
2	0.000	1.000	0.300	❌ NO
4	0.000	1.000	0.300	❌ NO
8	0.000	1.000	0.300	❌ NO
10	0.000	1.000	0.300	❌ NO

Result: ✅ **CORRECT CERTIFICATE SELECTED**
- Certificate: `0587-3024-5207-2897-6200`

- Address: `"32 Summerfields Drive, Blaxton, DONCASTER, DN9 3BG"`
- Final Score: `1.000`

Scoring Algorithm

Weights:

- **House Number Match**: 70%
- **Street Name Match**: 30%

House Number Scoring:

- **Exact match** (same number, no flat): `1.0`
- **Exact match** (same number, same flat): `1.0`
- **Partial match** (same number, different flat): `0.7`
- **Partial match** (same number, one has flat, other doesn't): `0.8`
- **Range match** (number falls within range): `0.6`
- **No match**: `0.0`

Street Name Scoring:

- Calculate: `(matching words) / (total words in target)`
- Words filtered: Only words > 2 characters
- Punctuation removed before comparison

Combined Score:

```
totalScore = (houseNumScore * 0.7) + (streetScore * 0.3)
```

Key Features

1. ✅ **Robust house number extraction** - Handles flats, ranges, letter suffixes
2. ✅ **Punctuation-agnostic matching** - Ignores commas, periods, etc.
3. ✅ **Comprehensive logging** - Easy to debug in production
4. ✅ **Defensive programming** - Handles edge cases gracefully
5. ✅ **Prioritizes house numbers** - 70% weight on exact house number match
6. ✅ **Top 5 summary** - Shows best matches for debugging

Files Modified

1. **src/utils/epcHandler.js**
   - Added `normalizeTextForMatching()` function
   - Enhanced `findBestAddressMatchFromScrapedData()` with logging
   - Enhanced `getCertificateNumber()` with defensive checks
   - Lowered score threshold from 0.4 to 0.3

2. **test-epc-matching-debug.js** (new)
   - Comprehensive test script for DN9 3BG postcode
   - Tests all 39 certificates
   - Validates house number extraction
   - Shows detailed scoring breakdown

Recommendations for Production

If wrong certificates are still selected in production:

1. **Check input data**:
   - Log the exact address string being passed
   - Verify postcode extraction is correct
   - Ensure address isn't empty/null

2. **Review upstream code**:
   - Check where address data comes from
   - Verify CSV parsing is working correctly
   - Ensure target address detection is accurate

3. **Monitor logs**:
   - Look for "⚠️ Falling back to first certificate" warnings
   - Check if score threshold is being met
   - Review top 5 matches to see if correct cert is even in results

4. **Postcode validation**:
   - Ensure postcode is being extracted correctly
   - Verify postcode format is correct (e.g., "DN9 3BG" not "DN93BG")

Branch Information

- **Branch**: `fix/epc-matching-debug`
- **Status**: ✅ Tested and working
- **Ready for**: Code review and PR creation

Next Steps

1. ✅ Test with DN9 3BG postcode (39 certificates) - **PASSED**
2. ✅ Verify correct certificate selection - **PASSED**
3. ✅ Add comprehensive logging - **DONE**
4. ✅ Create documentation - **DONE**
5. ⏳ Commit and push to GitHub
6. ⏳ Create pull request
7. ⏳ Test in production with Apify actor