Projektni seminar

# Razvoj eksperimentalnega sistema za simulacijo sodelovanja LLM-jev na napovednih trgih

Tehnična dokumentacija

Avtor: Marjan Meglen

Mentor: dr. Aleksandar Tošić

Somentor: dr. Niki Hrovatin

UP FAMNIT, Koper

Maj 2026

# Kazalo

# 1 Uvod

Ta dokument opisuje tehnično zgradbo sistema za izvajanje LLM napovedi na napovednih trgih. Poudarek je na strukturi modulov, vhodnih in izhodnih podatkih, načinu izvajanja eksperimentov, validaciji ter pripravi analitičnih izhodov.

# 2 Struktura sistema

Sistem je razdeljen na tri glavne mape:

| **Mapa** | **Vloga** |
|---|---|
| `data-collection/` | Zbiranje, filtriranje in izvoz vhodnih podatkov. |
| `llm-trader/` | Izvajanje LLM eksperimentov in shranjevanje napovedi. |
| `analysis/` | Statistična analiza rezultatov, grafi, tabele in poročila. |

## 2.1 Podatkovni pipeline

Podatkovni pipeline pripravi vhodne datoteke za eksperimentalni del. Glavne faze so:

1. zbiranje metapodatkov trgov,
2. zbiranje dnevnih tržnih verjetnosti,
3. indeksiranje novic iz več virov,
4. izbor relevantnih člankov z embeddingi,
5. prenos polnega besedila člankov,
6. semantično filtriranje z LLM modelom,
7. izvoz podatkov v obliko, ki jo uporablja `llm-trader`.

Končni izvoz podatkovnega dela so datoteke:

- `markets.json`,
- `daily_data.json`,
- `news.json`.

## 2.2 Eksperimentalni modul

Eksperimentalni modul se nahaja v `llm-trader/`. Vstopna točka je `run_experiment.py`. Logika je razdeljena na manjše module:

| Datoteka | Naloga |
|---|---|
| `load_data.py` | Naloži in validira vse vhodne JSON datoteke. |
| `types.py` | Definira podatkovne razrede za trge, novice, konfiguracije in napovedi. |
| `select_news.py` | Izbere novice, ki so vidne v trenutnem eksperimentalnem pogoju. |
| `build_prompt.py` | Sestavi končni poziv za LLM model. |
| `llm_client.py` | Vsebuje `mock` in `ollama` odjemalca. |
| `parse_llm_response.py` | Razčleni in validira odgovor modela. |
| `experiment_runner.py` | Izvede eksperiment po dnevih in shrani rezultate. |

# 3 Vhodni podatki

Vhodni podatki so v mapi `llm-trader/input_data/`. Vsaka datoteka je JSON tabela objektov.

## 3.1 `markets.json`

Datoteka vsebuje definicije napovednih trgov. Obvezna polja so:

- `market_id`: enolični identifikator trga,
- `question`: vprašanje trga,
- `resolution_date`: datum razrešitve v obliki `YYYY-MM-DD`.

Polje `resolved_yes` je opcijsko in privzeto nastavljeno na `true`.

## 3.2 `daily_data.json`

Datoteka vsebuje dnevne tržne verjetnosti. Vsaka vrstica ima:

- `market_id`,
- `date`,
- `probability_yes`,
- `probability_no`.

Sistem preveri, da sta verjetnosti med 0 in 1 ter da je njuna vsota enaka 1.

### 3.3 news.json

Novice so vezane na trg in datum. Obvezna polja so news_id, market_id, date, title in content. Datoteka lahko vsebuje tudi metapodatke iz semantičnega filtriranja, kot so rang podobnosti naslova, ocena relevantnosti in kratka razlaga relevantnosti.

### 3.4 instructions.json in configurations.json

instructions.json vsebuje osnovna navodila za model. configurations.json določa eksperimentalne pogoje. Konfiguracija vsebuje:

- identifikator konfiguracije,
- referenco na navodilo,
- logično vrednost show_market_probabilities,
- vrednost other_traders: llms, humans ali none,
- vrednost news_mode: current_day ali none.

## 4 Izvajanje eksperimenta

Eksperiment se zažene iz mape llm-trader/. Primer testnega zagona:

```
python3 run_experiment.py --market-id fed_decision_in_september \
  --configuration-id 1 --provider mock
```

Primer zagona z Ollama-kompatibilnim strežnikom:

```
python3 run_experiment.py --market-id fed_decision_in_september \
  --configuration-id 1 --provider ollama --model-name qwen3.5:4b
```

Glavni argumenti so:

| Argument | Opis |
|---|---|
| -market-id | Identifikator trga, za katerega se izvede eksperiment. |
| -configuration-id | Identifikator konfiguracije poziva. |
| -provider | mock za testiranje ali ollama za dejanski LLM klic. |
| -model-name | Ime modela pri uporabi ponudnika ollama. |
| -ollama-base-url | Naslov Ollama-kompatibilnega API strežnika. |
| -timeout-seconds | Časovna omejitev posameznega zahtevka. |

# 5 Gradnja poziva

Poziv se sestavi iz več delov:

1. osnovno navodilo za model,
2. vprašanje trga,
3. trenutni datum,
4. datum razrešitve,
5. opcijski opis drugih udeležencev,
6. opcijske trenutne tržne verjetnosti,
7. opcijske novice za trenutni dan,
8. zahteva za JSON odgovor.

Model mora vrniti objekt z obliko:

```
{
  "reasoning": "short explanation",
  "probability_yes": 0.0
}
```

# 6 Validacija in obravnava napak

Validacija se izvaja pred in po LLM klicu. Pred izvedbo sistem preveri obstoj datotek, JSON strukturo, obvezna polja, unikatnost identifikatorjev, veljavnost datumov, pravilne reference med datotekami in pravilnost verjetnosti.

Po LLM klicu sistem preveri, da je odgovor veljaven JSON objekt, da vsebuje polji `reasoning` in `probability_yes`, da je utemeljitev neprazen niz in da je verjetnost število med 0 in 1. Če pride do napake pri generiranju ali razčlenjevanju odgovora, se v mapo zagona zapiše diagnostična datoteka z napako, pozivom in neobdelanim odgovorom.

# 7 Izhodni podatki

Vsak zagon ustvari mapo:

`results/<market_id>_<configuration_id>_<YYYYMMDD_HHMMSS>/`

Glavna izhodna datoteka je `predictions.json`. Vsak zapis vsebuje:

- `market_id`,

- `configuration_id`,
- `date`,
- `reasoning`,
- `p_yes`,
- `p_no`,
- `rendered_prompt`,
- `complete_conversation`.

# 8 Analitični del

Analiza uporablja datoteke `predictions.json` iz mape `llm-trader/results/`. Glavni zvezek je `analysis/notebooks/preliminary_statistical_analysis.ipynb`. Izhodi so shranjeni kot:

- grafi v `analysis/outputs/figures/`,
- CSV tabele v `analysis/outputs/tables/`,
- LaTeX tabele in poročilo v `analysis/report/`.

V trenutnem obsegu analiza zajema 6 trgov, 12 konfiguracij in 20 dnevnih napovedi na kombinacijo trga in konfiguracije.

# 9 Vzdrževanje in razširjanje

Sistem je mogoče razširiti na več načinov:

- dodajanje novih trgov v `markets.json`,
- dodajanje novih dnevnih verjetnosti v `daily_data.json`,
- dodajanje novih virov novic v podatkovni pipeline,
- dodajanje novih konfiguracij pozivov,
- zamenjava LLM modela prek argumenta `-model-name`,
- razširitev analize z dodatnimi statističnimi merami.