1. Dokumentasjon for Avinor-konkurranse

Forfatter Adrian Fagerland **Dato** 29. september 2025 **Kontakt** adrian (at) example.com

1.1. Sammendrag

Denne dokumentasjonen beskriver løsningen utviklet til Avinors studentcase om å predikere sannsynlighet for samtidighet på norske AFIS-flyplasser i oktober 2025. Leveransen består av (1) dette dokumentet i PDF-format, (2) en CSV-fil med timevise prediksjoner for alle kombinasjoner av flyplassgrupper og timer i evalueringsperioden (`outputs/final_predictions.csv`), (3) studentbevis og (4) all kode på Github. Dokumentet følger strukturen i Vedlegg 3 og forklarer metodevalg, arkitektur, modeller og kodebase. Mye av denne informasjonen star også i kodebasens `README.md`.

1.2. Metodevalg og tilnærming

1.2.1. Problemforståelse

Målet er å estimere sannsynligheten for at minst to fly er i operasjonsvinduet samtidig («samtidighet») for hver time i oktober 2025. Caset krever en sannsynlighet mellom 0 og 1 og evalueres med AUC-ROC. Løsningen må håndtere usikkerhet i ruteplanen, forsinkelser og eksterne forhold som vær.

1.2.2. Overordnet strategi

1. Plan- og historikkbasert grunnlag bygger en deterministisk tidslinje for hver flyplassgruppe basert på publisert ruteplan og operasjonsvinduene fra caset (avgang −15/+8 minutter, landing −16/+5 minutter).
2. Monte Carlo-simuleringer kjøres 120–200 ganger per time der faktiske avgangs- og ankomsttider perturberes med historiske forsinkelser og kanselleringer. Dette skaper sannsynlighetsfordelinger for samtidighet og gir robuste features.
3. Tabulær sannsynlighetsmodell trener en gradient-boosted beslutningstrær-modell (LightGBM) på et stort feature-sett som kombinerer planlagte tellinger, simuleringstatistikk, historie, vær og kalenderinformasjon.
4. Stacking og ensembling evaluerer et panel av modeller (logistisk regresjon, MLP, LightGBM, CatBoost, XGBoost) med tidsbasert kryssvalidering og velger den beste modellen (LightGBM) med mulighet for logistisk stacking dersom det gir gevinst.

Denne kombinasjonen gir forklarbarhet (plan/feature-baserte signaler) og robusthet mot usikkerhet (simuleringer). LightGBM ble valgt fordi den ga høyest OOF-AUC (0.979) samtidig som den håndterer heterogene features uten omfattende skalering.

1.2.3. Dataflyt

1. Les inn historiske flight-logger (`data/historical/`), ruteplan for oktober (`data/schedule/`), og Avinors `preds_mal.csv` som mal for nøklene.
2. Berik datasettet med eksterne kilder som værdata (MET APIs), kalender (helligdager og skoleferier) og metadata om flyplassgrupper (`data/airportgroups.csv`).
3. Generer minuttoppløst samtidighetsmatrise og features per time via `src/avinor/feature_engineering.py`.
4. Kjør simuleringer og aggreger statistikk via `src/avinor/simulation.py`.
5. Tren modeller, lagre mellomresultater i `outputs/feature_cache/`, og produser endelige prediksjoner i `outputs/final_predictions.csv`.

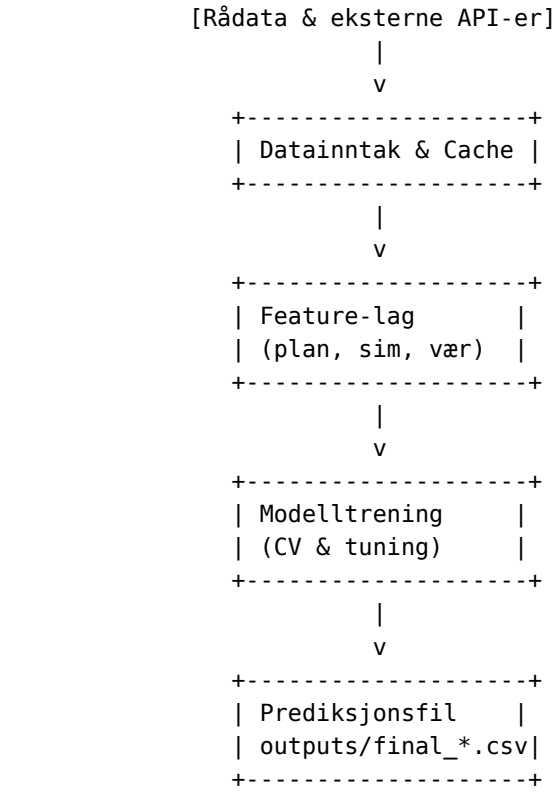
1.3. Systemstruktur og arkitektur

1.3.1. Komponenter

1. Datainntak (`scripts/fetch_*.py`, `src/avinor/data_loader.py`) leser case-data og henter eksterne kilder før mellomlagring i `data/`.
2. Feature-laget (`src/avinor/feature_engineering.py`, `src/avinor/external_features.py`, `src/avinor/tabular.py`) bygger baseindeks, utleder samtidighetsfeatures og kombinerer vær- og kalenderinformasjon.
3. Simuleringen (`src/avinor/simulation.py`, `src/avinor/models/delay_cancellation.py`) gjennomfører en Monte Carlo-prosess som vurderer sannsynlige forsinkelser og kanselleringer.
4. Modelltreningen (`src/avinor/pipeline.py`) utfører tidsbasert kryssvalidering, hyperparametertuning og modellpersistens.
5. Prediksjon og levering (`scripts/run_pipeline.py`) genererer `outputs/final_predictions.csv` med korrekt format.

1.3.2. Arkitekturoversikt

Figur 1. Systemarkitektur



Systemet er modulært, hver blokk kan kjøres uavhengig og cacher resultater (f.eks. simuleringer i `outputs/feature_cache/`) slik at senere kjøringer gjenbruker arbeid. Alle skript tar kommandolinjeflagg for logging og frøsetting.

1.4. Modeller og algoritmer

1.4.1. Featurefamilier

- Planbaserte features inkluderer tellinger av planlagte avganger/ankomster i ±30/60/90 minutter, forrige time, forholdstall og deterministisk samtidighet beregnet som differanse-array.

- Simuleringsfeatures beskriver forventning og variasjon i samtidighet, sannsynlighet for minst én overlapp, arealet under samtidighetskurven, minutter med ≥2 eller ≥3 samtidigheter samt minimumsgap.
- Historiske basisrater gir gjennomsnittlig samtidighet per gruppe×time, måned×time og ukedag×time som priors.
- Nær-tidsdynamikk bruker rullerende middel og standardavvik (24/72/168/336 timer), forrige time og «hours since positive» uten fremtidslekkasje.
- Værsignaler dekker vindstyrke, vindkast, nedbør, skyhøyde og binære indikatorer for IFR-forhold, torden, snø og regn.
- Kalender og syklikalitet omfatter helligdager, skoleferier (heuristikk), sykliske encodinger for time, ukedag og måned samt sesong.

1.4.2. Modellpanel

Alle modeller trenes via tidsbasert kryssvalidering (`temporal_cv_indices`) der treningsfold er historiske måneder før valideringsmåneden.

Modell	Viktige hyperparametre	OOF-AUC

LightGBM (valgt)	`learning_rate=0.04`
	`num_leaves=80`
	`min_child_samples=45`
	`subsample=0.85`
	`colsample_bytree=0.8`
	`reg_alpha=0.05`
	`reg_lambda=0.4`
		0.979

CatBoost	`learning_rate=0.04`
	`depth=7`
	`l2_leaf_reg=6.0`
	`bagging_temperature=0.3`
		0.979

XGBoost	`learning_rate=0.05`
	`max_depth=6`
	`subsample=0.85`
	`colsample_bytree=0.75`
	`reg_lambda=2.0`
		0.979

MLP	To-lags MLP (128–64) med ReLU, batchnorm og dropout 0.2	0.976

Logistisk regresjon	`C=0.1`
	`penalty="l2"`
	`class_weight="balanced"`
		0.971

LightGBM ble valgt som finalemodell (`final_overall_auc=0.9788`). Ensemblet ga lavere total AUC, men brukes som analyseverktøy. Modellartefakter og out-of-fold prediksjoner lagres i `outputs/models/` og `outputs/feature_cache/`.

1.4.3. Evalueringsoppsett

Evalueringsopplegget bygger hele sitt rammeverk på AUC-ROC, slik caset beskriver. Rullerende månedsbaserte fold (fire siste måneder i treningsdata) sørger for at hver valideringsfold trener på alle måneder før valideringsmåneden og testes på en ny måned. Med denne strategien oppnår LightGBM fold-AUC = 0.966, 0.981, 0.984 og 0.984, mens hold-out for juli 2025 ender på 0.984.

Simuleringsmodellen alene treffer 0.955 for juli og en enkel logistisk baseline 0.978. Feature-importance peker på `feat_sched_concurrence` (59.4 % av gain), `sim_prob_any_overlap_mean` (7.2 %) og `sim_overlap_minutes_ge3_mean` (4.8 %) som de mest dominerende signalene, dokumentert i `outputs/analysis/lgbm_analysis.json`.

1.5. Kildekode, installasjon og kjøring

1.5.1. Kodeoversikt

- `src/avinor/` er hovedbiblioteket med datalastere, featurebyggere, simuleringer og modellpipeline.
- `scripts/` samler kjørbare oppgaver (fetch, simulering, pipeline).
- `data/` inneholder casefiler og eksterne data (rå og prosesserte).
- `outputs/` lagrer modellartefakter, mellomlagring (`feature_cache/`) og sluttprediksjoner.
- `tests/` rommer eksempeltester for sentrale komponenter.

1.5.2. Miljø og avhengigheter

Prosjektet bruker `uv` for repeterbar avhengighetsstyring. Python >=3.11 kreves.

```
uv sync
```

Dette installerer både kjøre- og utviklingsavhengigheter som definert i `pyproject.toml` og `uv.lock`.

1.5.3. Datatilrettelegging

1. Kopier alle casefiler til `data/` (inkludert `airportgroups.csv`, historikk og malfiler).
2. Sett miljøvariabler ved behov (f.eks. `FROST_CLIENT_ID` for MET Frost API).
3. Kjør datascripts (valgfrie dersom data allerede ligger lokalt)

```
uv run scripts/fetch_airport_metadata.py --mapping-path data/airportgroups.csv
uv run scripts/fetch_met_weather.py
FROST_CLIENT_ID=... uv run scripts/fetch_met_weather_history.py
uv run scripts/generate_calendar_events.py
```

1.5.4. Kjøre pipeline

```
uv run scripts/run_simulation_and_ensemble.py --only-simulations
uv run scripts/run_pipeline.py --log-level INFO
```

Dette produserer `outputs/final_predictions.csv`. Filen følger casets formatkrav (UTF-8, komma-separert, tre desimaler, punktum som desimaltegn). Dersom en ny ruteplan mottas, bytt filen i `data/` og kjør skriptene på nytt.