OWNML MACHINE LEARNING CANVAS

Designed for: Retail Operation | Designed by: Bishakha, Javaid & Yesha | Date: 08/08/2025 | Iteration: 2

PREDICTION TASK

What is the type of task?

Object detection + downstream classification (computer vision). Detect & classify product instances in shelf images, then aggregate counts to infer shelf status.

Which entity are predictions made on?

Shelf segments / product SKUs within shelf images (detected product instances).

What are the possible outcomes to predict?

For each SKU or shelf segment — In-stock (sufficient count) or Out-of-stock / low-stock (below threshold). Also raw outputs: bounding boxes, class labels, and counts per SKU.

When are outcomes observed?

Outcomes observed at image-capture time (camera snapshots). Ground-truth labels come from annotated images (YOLO-style .txt label files). Observed outcomes for evaluation are the labeled images in the dataset.

DECISIONS

How are predictions turned into actionable recommendations or decisions for the end-user? (Mention parameters of the process / application for this.)

-If aggregated counts for a shelf segment fall below a configured threshold → trigger a restock alert to store staff or create a work order in the store operations dashboard.

If repeated OOS detected across the same SKU/locations → escalate to category manager / inventory team (possible re-order or planogram change).

Parameters:

count threshold per SKU, minimum confidence score for detection, cooldown/time-window (e.g., suppress repeat alerts for X minutes), and aggregation window (instant vs. hourly).

Example flow:

Camera → model inference → aggregate counts → compare to threshold → API call to notify staff.

VALUE PROPOSITION

Who is the end beneficiary, and what specific pain points are addressed?

End beneficiary & pain points:

Store managers, shelf replenishment teams, inventory planners. Pain points addressed: missed sales due to out-of-stock, inefficient manual shelf checks, time spent walking aisles for checks, shrink/untracked product movement.

How will the ML solution integrate with their workflow, and through which user interfaces?

Integration & UI:

Integrates with store ops dashboard, mobile push/Slack/SMS alerts for floor staff, or a backend API that creates tickets/tasks. Optionally a web UI showing shelf images with detections and OOS flags.

DATA COLLECTION

How is the initial set of entities and outcomes sourced (e.g., database extracts, API pulls, manual labeling)?

Initial dataset source:

Labeled shelf images (images + YOLO-style .txt label files) — e.g., the notebook uses a dataset retail-product-checkout-35 and local folders of images/labels. Labels produced by human annotators (bounding boxes & class ids).

What strategies are in place to update data continuously while controlling cost and maintaining freshness?

Update strategy:

Periodic data pull from store cameras (daily/hourly). Use automated ingestion pipeline to collect new images, sample and label (semi-automated / active learning) to control cost. Retrain when new SKUs or distribution shift > threshold or quarterly. Use small human-in-the-loop correction batches for noisy predictions.

DATA SOURCES

Where can we get data on entities and observed outcomes? (Mention internal and external database tables or API methods.)

Where to get data:

Internal: store camera image feeds (S3 / blob storage / internal file server), annotated image repo used during training (images/, labels/ folders).

External:

Public retail product imagery datasets only if needed for augmentation; synthetic augmentation (image transforms) used in notebook. Local dataset directory and label files (.txt) as data source.

Link:

https://universe.roboflow.com/cdio-zmfmj/retail-product-checkout

IMPACT SIMULATION

What are the cost/gain values for (in)correct decisions?

Cost/gain values: (assumptions — change if you have actual numbers)

Cost of missed sale per sold-out instance: $X (lost margin).

Cost of unnecessary restock visit: $Y (staff time).

Which data is used to simulate pre-deployment impact?

Historic POS data joined to shelf-image timestamps to estimate sales lost during observed OOS windows (if available).

What are the criteria for deployment?

Model **mAP50 ≥ 0.9** for key SKUs and per-class recall above acceptable threshold. **Business criterion:** simulated net gain (revenue recovered − operational cost) > 0.

Are there fairness constraints?

Fairness constraints: Ensure model is robust across store lighting, shelf angles, and product packaging variants; avoid systematically missing specific SKUs (e.g., dark-packaged items).

MAKING PREDICTIONS

Are predictions made in batch or in real time?

Batch or real-time: Near real-time inference at edge (camera/edge device) or periodic batch (e.g., every 5–60 minutes) depending on edge capacity. Notebook implies training locally and inference per-image (fits real-time use).

How frequently?

Frequency: Could be continuous (every image) or aggregated hourly. Start with per-image inference and aggregate to a sliding window.

How much time is available for this (including featurization and decisions)?

Latency budget: Per-image inference should be sub-second to a few seconds on edge GPU/accelerator (notebook used GPU during training). If running on CPU, expect longer latencies, plan for dedicated inference hardware for real-time.

Which computational resources are used?

Trained using GPU (notebook logs show GPU memory ~2.2G). Inference options: edge GPUs (NVIDIA Jetson/edge TPU alternatives) or cloud GPUs for centralized inference.

BUILDING MODELS

How many models are needed in production?

One main object detector model to detect all product classes; optionally a small post-processing model (rule-based) to decide OOS based on counts, time-of-day, and confidence. If multiple store formats, consider a single generalizable model.

When should they be updated?

Retrain when the dataset grows substantially, monthly/quarterly, or after distribution shift detected. Also perform quick fine-tuning for new SKUs.

How much time is available for this (including featurization and analysis)?

Notebook trains for multiple epochs (e.g., 75 epochs shown). Practically, retrain in batch overnight, fine-tune incrementally for faster updates.

Which computation resources are used?

GPU for training ((NVIDIA GeForce 940MX, 4096MiB).

FEATURES

What representations are used for entities at prediction time?

Representations at prediction time: Raw RGB images. Post-processing representation: bounding boxes + class labels + confidences + aggregated counts per SKU + temporal smoothing features (counts in last N frames).

What aggregations or transformations are applied to raw data sources?

Aggregations/transformations: Convert detection boxes → counts per SKU per shelf segment, apply confidence threshold, non-maximum suppression already in the detector. Compute moving averages / dwell-time filters to remove noise.

MONITORING

Which metrics and KPIs are used to track the ML solution's impact once deployed, both for end-users and for the business?

Model-level: mAP50, per-class precision & recall, false positive rate, false negative rate, inference latency, confidence distribution.

Business-level: OOS rate (before/after), alerts triggered, true positive alerts (validated by staff), restock response time, recovered sales / revenue uplift, number of unnecessary visits.

How often should they be reviewed?

Review cadence: Model metrics — daily/weekly dashboards; business KPIs — weekly / monthly. Trigger retraining or investigation on sudden drops (e.g., 10% drop in mAP or per-class recall).