**Suggested Drift Dashboard Layout**

**Metrics to monitor:**

1. Missing values ✅
2. Data distributions ✅
3. Outliers ✅
4. Schema/variables ✅
5. Feature correlations
6. Target distribution (if available)
7. Prediction distribution
8. Statistical distances
9. Concept drift classifier score
10. Data volume trends

## Streamlit Drift Dashboard Layout

### ****Sidebar (Controls)****

* Upload baseline.csv and current.csv
* Choose feature(s) to analyze
* Select metrics (checkboxes: Missing Values, Distribution Drift, Outliers, etc.)

### ****Main Tabs****

1. **📊 Overview**
   * Summary table: % missing, # features, # rows, # outliers, data volume change
   * High-level alerts (red/yellow/green flags)
2. **📉 Data Distribution Drift**
   * Histograms/KDE plots (baseline vs current)
   * PSI, KS-test, Chi-square results
3. **❗ Outlier Analysis**
   * Boxplots of features
   * % of outliers baseline vs current
4. **📑 Schema & Missing Values**
   * Table: Expected vs Actual variables
   * Missing % baseline vs current
5. **🔗 Feature Correlation Drift**
   * Correlation heatmaps baseline vs current
   * Difference heatmap (absolute delta correlations)
6. **🤖 Prediction Drift** (if predictions uploaded)
   * Predicted probability distributions baseline vs current
   * Chi-square/JS Divergence
7. **⚠️ Concept Drift Detection**
   * Train a classifier to separate baseline vs current → show accuracy
   * High accuracy → indicates drift

## ⚡ Suggested Implementation Flow

1. **Backend Drift Functions** (Python functions for each metric)
2. **Streamlit UI** → Sidebar (file upload + options)
3. **Visualization** → Plots/tables for each drift metric
4. **Alert System** → Color coding / thresholds (e.g., PSI > 0.2 = red flag)

## 📝 Next Step

I can give you a **starter Streamlit code skeleton** that:

* Uploads baseline & current CSV
* Computes **the first 4 metrics (missing, distribution, outliers, schema)**
* Displays them in nice Streamlit tables/plots

Then we can **incrementally add** the advanced ones (correlation drift, prediction drift, concept drift).

👉 Do you want me to write the **initial Streamlit skeleton code** for the first 4 drift metrics right now?

##### You said:

give me the full code know?

###### ChatGPT said:

Thought for 23s

Streamlit Drift Dashboard· python

from typing import Tuple, Dict

st.pyplot(fig2, clear\_figure=True)

st.markdown("---")

if not corr\_b.empty and not corr\_c.empty:

# Align and compute absolute delta

common\_idx = corr\_b.index.intersection(corr\_c.index)

corr\_b2 = corr\_b.loc[common\_idx, common\_idx]

corr\_c2 = corr\_c.loc[common\_idx, common\_idx]

delta = (corr\_c2 - corr\_b2).abs()

fig3, ax3 = plt.subplots(figsize=(6, 5))

heatmap(ax3, delta, 'Absolute Correlation Δ')

st.pyplot(fig3, clear\_figure=True)

# =============================

# 6) PREDICTION DRIFT

# =============================

if show\_pred:

with tabs[5]:

st.subheader("Prediction Distribution Drift")

if opt\_pred\_col and (opt\_pred\_col in baseline.columns) and (opt\_pred\_col in current.columns):

pb = baseline[opt\_pred\_col]

pc = current[opt\_pred\_col]

jsd = js\_divergence(pb, pc)

st.write(f"Jensen–Shannon divergence (squared) between prediction distributions: \*\*{jsd:.4f}\*\*")

fig, ax = plt.subplots(figsize=(8, 4))

if is\_numeric(pb):

ax.hist(pd.to\_numeric(pb, errors='coerce').dropna(), bins=30, alpha=0.5, label='baseline', density=True)

ax.hist(pd.to\_numeric(pc, errors='coerce').dropna(), bins=30, alpha=0.5, label='current', density=True)

ax.set\_xlabel(opt\_pred\_col)

ax.set\_ylabel('Density')

ax.set\_title('Prediction Distribution')

ax.legend()

else:

bvc = pb.astype(str).value\_counts(normalize=True)

dvc = pc.astype(str).value\_counts(normalize=True)

cats = sorted(list(set(bvc.index) | set(dvc.index)))

bvals = [bvc.get(k, 0) for k in cats]

dvals = [dvc.get(k, 0) for k in cats]

x = np.arange(len(cats))

ax.bar(x - 0.2, bvals, width=0.4, label='baseline')

ax.bar(x + 0.2, dvals, width=0.4, label='current')

ax.set\_xticks(x)

ax.set\_xticklabels(cats, rotation=90)

ax.set\_ylabel('Proportion')

ax.set\_title('Prediction Category Proportions')

ax.legend()

st.pyplot(fig, clear\_figure=True)

else:

st.info("Provide a valid prediction column name in the sidebar to enable this tab.")

# =============================

# 7) CONCEPT DRIFT

# =============================

if show\_concept:

with tabs[6]:

st.subheader("Concept Drift (baseline vs current classifier AUC)")

auc = concept\_drift\_auc(baseline, current)

if pd.isna(auc):

st.info("Not enough data or incompatible features to compute concept drift AUC.")

else:

st.metric(label="Drift Classifier ROC-AUC", value=f"{auc:.3f}", delta=None)

st.caption("Higher AUC indicates the model can easily tell baseline vs current → stronger drift.")

else:

st.title("📊 Drift Monitoring Dashboard")

st.write("Upload a \*\*Baseline CSV\*\* and a \*\*Current CSV\*\* in the sidebar to begin.")

st.markdown("---")

st.caption("Tip: Start with the 'Schema & Missing' and 'Distribution Drift' tabs. Use PSI > 0.2 or KS/Chi2 p-value < 0.05 as strong drift signals.")

Done! I’ve dropped a full, single-file **Streamlit drift monitoring dashboard** into the canvas (named **“Streamlit Drift Dashboard”**). It:

* Uploads baseline.csv & current.csv
* Computes: PSI, KS/Chi-square, Jensen–Shannon, Wasserstein, outlier rate drift, schema diffs, missing-value drift
* Shows correlation drift heatmaps
* (Optional) Prediction drift if you specify a prediction column
* Concept drift via a baseline-vs-current classifier ROC-AUC
* Nice tabs + selectable feature plots

### How to run

pip install streamlit pandas numpy scipy scikit-learn matplotlib

streamlit run streamlit\_drift\_dashboard.py

If you want me to add **alerts + thresholds** (e.g., PSI>0.2 turns red, Slack/email hooks) or wire a **“Trigger Retraining”** button to SageMaker, say the word and I’ll extend the code.