Reg

December 5, 2025

[29]:
```
# ========================================
# CHUNK 0: MASTER SETUP & IMPORTS
# ========================================
import pandas as pd
import numpy as np
import time
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from statsmodels.stats.outliers_influence import variance_inflation_factor

# CONFIGURATION
# Set to None to use all data (Production), or a small number (e.g., 50000) for␣
 ↪fast testing
SAMPLE_SIZE = None

print("--- 1. Loading Data ---")
df_flights = pd.read_csv('US_flights_2023.csv', low_memory=False)
df_airports = pd.read_csv('airports_geolocation.csv')
df_weather = pd.read_csv('weather_meteo_by_airport.csv')

print("--- 2. Merging & Prepping ---")
# Convert dates
df_flights['FlightDate'] = pd.to_datetime(df_flights['FlightDate'])
df_weather['time'] = pd.to_datetime(df_weather['time'])

# SAMPLING (Crucial for speed if dataset is huge)
if SAMPLE_SIZE and len(df_flights) > SAMPLE_SIZE:
    print(f"   -> Sampling {SAMPLE_SIZE} rows...")
    df_flights = df_flights.sample(n=SAMPLE_SIZE, random_state=42)
```

```
# MERGE AIRPORTS (Get Origin_State)
df_merged = df_flights.merge(
    df_airports[['IATA_CODE', 'STATE']],
    left_on='Dep_Airport',
    right_on='IATA_CODE',
    how='left'
)
df_merged = df_merged.rename(columns={'STATE': 'Origin_State'}).
 ↪drop(columns=['IATA_CODE'])

# MERGE WEATHER (Get Rain/Wind)
df_final = df_merged.merge(
    df_weather[['airport_id', 'time', 'prcp', 'snow', 'wspd', 'tavg']],
    left_on=['Dep_Airport', 'FlightDate'],
    right_on=['airport_id', 'time'],
    how='left'
)
# Cleanup
df_final = df_final.drop(columns=['airport_id', 'time'])

print(f"SETUP COMPLETE. 'df_final' is ready. Shape: {df_final.shape}")
```

```
--- 1. Loading Data ---
--- 2. Merging & Prepping ---
SETUP COMPLETE. 'df_final' is ready. Shape: (6743404, 29)
```

[30]:
```
# ========================================
# CHUNK 1: Feature Selection & Split
# ========================================
print("--- Setting up Final Regression Data ---")

# 1. FINAL FEATURES
features_final = [
    'Dep_Delay',          # Primary driver
    'Flight_Duration',    # Proxy for Distance
    'Day_Of_Week',
    'DepTime_label',      # Morning/Evening
    'Airline',
    'Aicraft_age',
    'Origin_State',
    'tavg', 'prcp', 'wspd' # Weather factors
]
target = 'Arr_Delay'

# 2. FILTER & CLEAN
df_reg = df_final[features_final + [target]].copy()
```

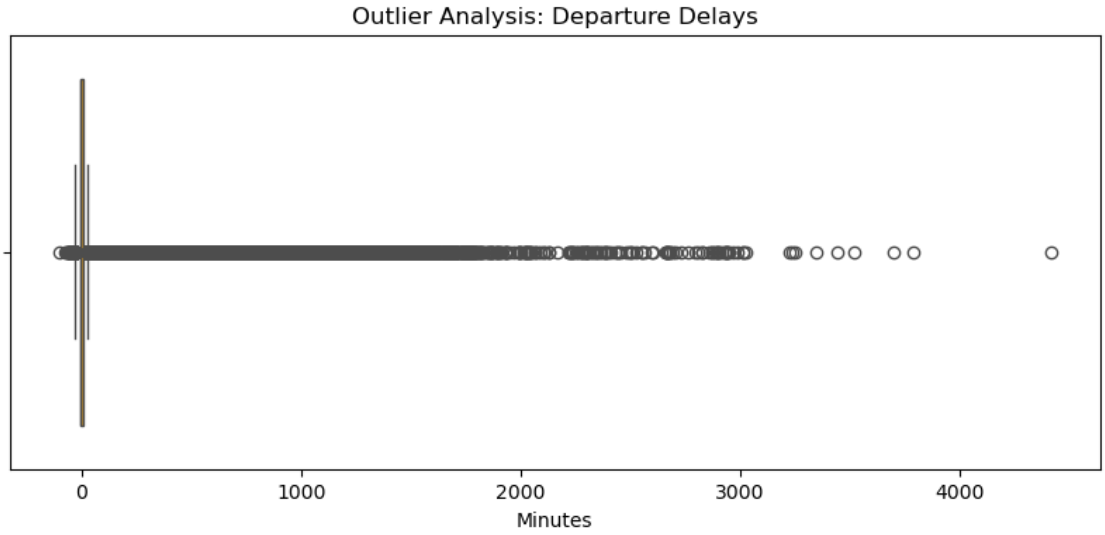
```
df_reg = df_reg.dropna()

# 3. SAMPLING FOR MODELING
SAMPLE_N = SAMPLE_SIZE
if SAMPLE_N and len(df_reg) > SAMPLE_N:
    df_reg = df_reg.sample(SAMPLE_N, random_state=42)

# 4. SPLIT
X = df_reg.drop(columns=[target])
y = df_reg[target]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

print(f"Data Ready. Training on {len(X_train)} rows, Testing on {len(X_test)}␣
 ↪rows.")
```

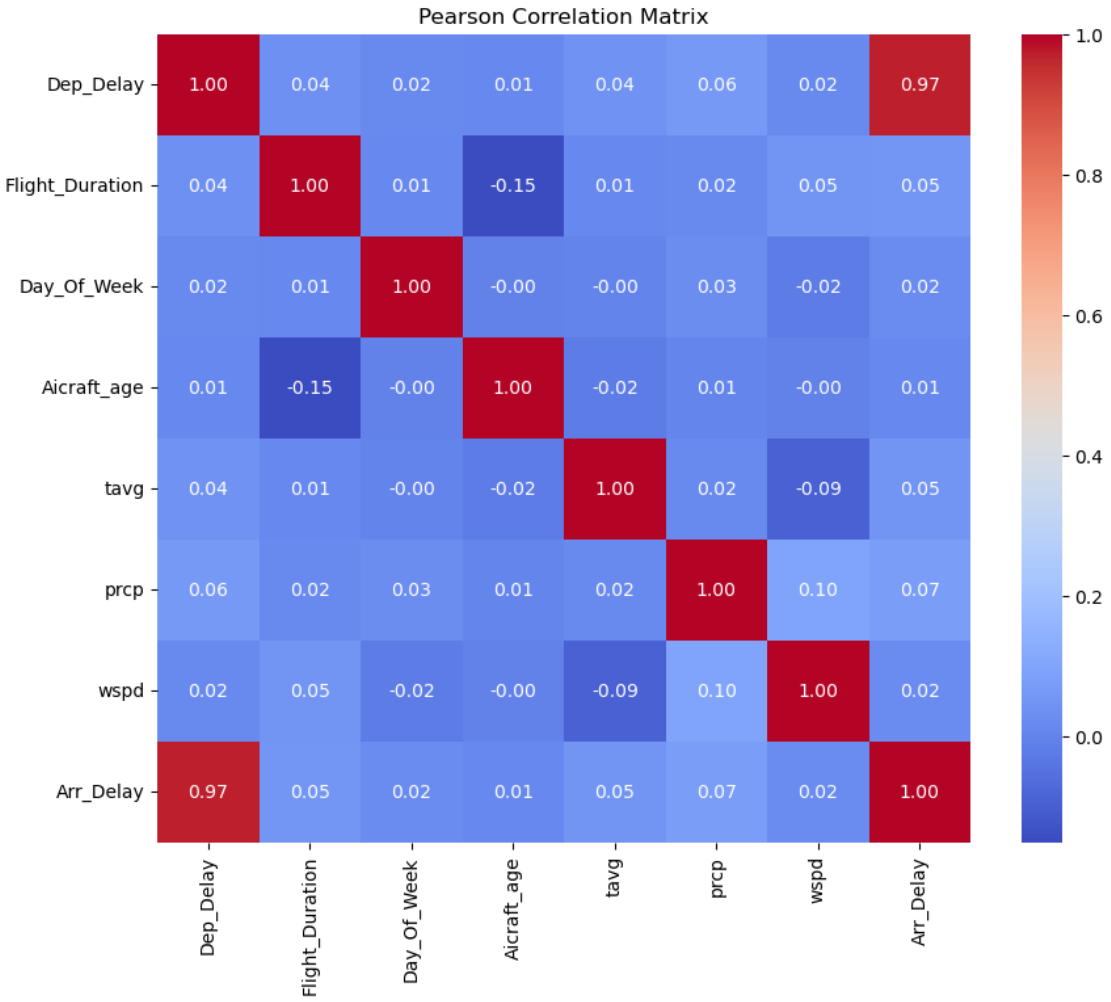
```
--- Setting up Final Regression Data ---
Data Ready. Training on 5394723 rows, Testing on 1348681 rows.
```

[31]:
```
# ========================================
# CHUNK 1.5: Phase I - Cleaning & EDA Audit
# ========================================
print("--- Phase I: Data Cleaning & EDA Audit ---")

# 1. DUPLICATE CHECK
initial_len = len(df_reg)
df_reg = df_reg.drop_duplicates()
print(f"Duplicates Removed: {initial_len - len(df_reg)}")

# 2. MISSING DATA CHECK
print(f"Missing Values Status: {df_reg.isna().sum().sum()} (Cleaned via␣
 ↪DropNA)")

# 3. OUTLIER ANALYSIS (Physics Check)
impossible_flights = df_reg[df_reg['Flight_Duration'] <= 0]
if len(impossible_flights) > 0:
    df_reg = df_reg[df_reg['Flight_Duration'] > 0]
    print(f" -> Removed {len(impossible_flights)} impossible flights (Duration␣
 ↪<= 0).")

# Visualization (Boxplot)
plt.figure(figsize=(10, 4))
sns.boxplot(x=df_reg['Dep_Delay'], color='orange')
plt.title('Outlier Analysis: Departure Delays')
plt.xlabel('Minutes')
plt.show()
```

```
# 4. COLLINEARITY CHECK (Correlation Matrix)
plt.figure(figsize=(10, 8))
numeric_df = df_reg.select_dtypes(include=['float64', 'int64'])
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Pearson Correlation Matrix')
plt.show()

# 5. VIF ANALYSIS
print("\n--- Syllabus Check: VIF Analysis ---")
X_vif = numeric_df.drop(columns=['Arr_Delay'], errors='ignore')
X_vif['intercept'] = 1

vif_data = pd.DataFrame()
vif_data["feature"] = X_vif.columns
vif_data["VIF"] = [variance_inflation_factor(X_vif.values, i) for i in␣
 ↪range(len(X_vif.columns))]
print(vif_data.sort_values('VIF', ascending=False))
```

```
--- Phase I: Data Cleaning & EDA Audit ---
Duplicates Removed: 492
Missing Values Status: 0 (Cleaned via DropNA)
 -> Removed 1 impossible flights (Duration <= 0).
```






```
--- Syllabus Check: VIF Analysis ---
           feature        VIF
7        intercept  21.607212
1  Flight_Duration   1.027907
3      Aicraft_age   1.024293
6             wspd   1.022501
5             prcp   1.015051
4             tavg   1.012356
0        Dep_Delay   1.008057
2      Day_Of_Week   1.002016
```

[32]:
```python
# ========================================
# CHUNK 1.6: Detailed Outlier Audit Function
# ========================================
def outlier_audit(df, features):
```

```
    num_feats = [c for c in features if pd.api.types.is_numeric_dtype(df[c])]
    cat_feats = [c for c in features if c not in num_feats]

    report_rows = []
    for c in num_feats:
        col = df[c]
        q1, q3 = col.quantile(0.25), col.quantile(0.75)
        iqr = q3 - q1
        lb, ub = q1 - 1.5 * iqr, q3 + 1.5 * iqr
        z = (col - col.mean()) / (col.std(ddof=0) if col.std(ddof=0) != 0 else␣
↪1)

        report_rows.append({
            "feature": c, "dtype": str(col.dtype), "count": len(col),
            "min": float(col.min()), "mean": float(col.mean()), "max":␣
↪float(col.max()),
            "outliers_1.5IQR": int(((col < lb) | (col > ub)).sum()),
            "outliers_z>4": int((z.abs() > 4).sum())
        })

    display(pd.DataFrame(report_rows).set_index('feature'))

    # Flags for specific domain rules
    flags = pd.Series(False, index=df.index)
    if 'Flight_Duration' in df.columns: flags |= (df['Flight_Duration'] <= 0)
    if 'Aicraft_age' in df.columns: flags |= (df['Aicraft_age'] > 80)
    if 'prcp' in df.columns: flags |= (df['prcp'] > 500)
    if 'wspd' in df.columns: flags |= (df['wspd'] > 200)
    if 'tavg' in df.columns: flags |= (df['tavg'] < -80) | (df['tavg'] > 60)

    return {"flag_index": df.index[flags]}

# Run the audit
audit = outlier_audit(X, features_final)

# Clean based on audit
X_clean = X.drop(index=audit['flag_index'])
y_clean = y.drop(index=audit['flag_index'])
print(f"Audit Complete. Rows removed: {len(audit['flag_index'])}")
```

feature	dtype	count	min	mean	max	outliers_1.5IQR	\
Dep_Delay	int64	6743404	-99.0	12.200987	4413.0	863168	
Flight_Duration	int64	6743404	0.0	140.297779	795.0	340654	
Day_Of_Week	int64	6743404	1.0	3.982793	7.0	0	
Aicraft_age	int64	6743404	1.0	13.480626	57.0	4248	
tavg	float64	6743404	-39.1	16.810826	42.2	24127	

```
prcp              float64  6743404   0.0     2.437808    571.5          1413666
wspd              float64  6743404   0.0    12.476340     59.4           146967

                  outliers_z>4
feature
Dep_Delay                52533
Flight_Duration          11680
Day_Of_Week                  0
Aicraft_age               4248
tavg                      2368
prcp                     80973
wspd                     11083

Audit Complete. Rows removed: 134
```

[33]:
```python
# ========================================
# CHUNK 2: The Pipeline Setup
# ========================================
# Identify columns
num_cols = X.select_dtypes(include=np.number).columns.tolist()
cat_cols = X.select_dtypes(exclude=np.number).columns.tolist()

# Transformer
preprocessor = ColumnTransformer(
    transformers=[
        ('num', 'passthrough', num_cols),
        ('cat', OneHotEncoder(handle_unknown='ignore', sparse_output=False),␣
 ↪cat_cols)
    ]
)

print("Pipeline defined.")
print(f"Numeric: {num_cols}")
print(f"Categorical: {cat_cols}")

# Quick Check on Delay Balance
delay_threshold = 15
print(f"Train Delayed > 15min: {(y_train > delay_threshold).sum()}")
print(f"Test Delayed > 15min:  {(y_test > delay_threshold).sum()}")
```

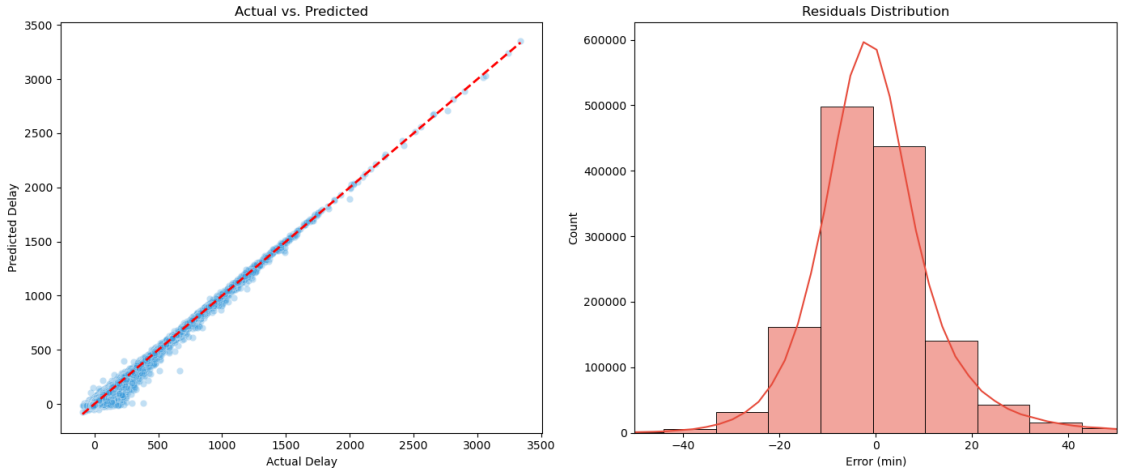
```
Pipeline defined.
Numeric: ['Dep_Delay', 'Flight_Duration', 'Day_Of_Week', 'Aicraft_age', 'tavg',
'prcp', 'wspd']
Categorical: ['DepTime_label', 'Airline', 'Origin_State']
Train Delayed > 15min: 1072119
Test Delayed > 15min:  267558
```

[34]:
```
# ========================================
# CHUNK 3: Model Training (Machine Learning)
# ========================================
def train_and_score(name, model):
    print(f"\nTraining {name}...")
    start = time.time()
    pipe = Pipeline(steps=[('preprocessor', preprocessor), ('model', model)])
    pipe.fit(X_train, y_train)
    preds = pipe.predict(X_test)

    rmse = np.sqrt(mean_squared_error(y_test, preds))
    r2 = r2_score(y_test, preds)
    print(f"  -> RMSE: {rmse:.4f} min")
    print(f"  -> R2:   {r2:.4f}")
    return {'RMSE': rmse, 'R2': r2, 'Model': pipe}

# Train Baseline
results = {}
results['Linear'] = train_and_score("Linear Regression", LinearRegression())
```

```
Training Linear Regression…
  -> RMSE: 13.5155 min
  -> R2:   0.9436
```

[35]:
```
# ========================================
# CHUNK 4: Diagnostics (Visuals)
# ========================================
final_model_pipe = results['Linear']['Model']
final_model_obj = final_model_pipe.named_steps['model']

y_pred_final = final_model_pipe.predict(X_test)
residuals = y_test - y_pred_final

plt.figure(figsize=(14, 6))

# Plot 1: Actual vs Predicted
plt.subplot(1, 2, 1)
sns.scatterplot(x=y_test, y=y_pred_final, alpha=0.3, color='#3498db')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--',␣
 ↪lw=2)
plt.title('Actual vs. Predicted')
plt.xlabel('Actual Delay'); plt.ylabel('Predicted Delay')

# Plot 2: Residuals
plt.subplot(1, 2, 2)
sns.histplot(residuals, kde=True, color='#e74c3c', bins=50)
```

```
plt.title('Residuals Distribution')
plt.xlabel('Error (min)'); plt.xlim(-50, 50)

plt.tight_layout()
plt.show()

print(f"Model R2: {r2_score(y_test, y_pred_final):.4f}")
```




```
Model R2: 0.9436
```

[36]:
```
# ==========================================
# CHUNK 5: Phase II - Statistical Inference & Stepwise Regression
# ==========================================
print("--- Phase II: Statistical Analysis (Syllabus Requirements) ---")

# 1. PREP DATA
STATS_SAMPLE = 100000
df_stats = df_reg.sample(n=min(len(df_reg), STATS_SAMPLE), random_state=42).
 ↪copy()
df_stats_encoded = pd.get_dummies(df_stats, columns=cat_cols, drop_first=True,␣
 ↪dtype=int)

y_sm = df_stats_encoded['Arr_Delay'].astype(float)
X_sm = df_stats_encoded.drop(columns=['Arr_Delay']).astype(float)
X_sm = sm.add_constant(X_sm)

# 2. STEPWISE REGRESSION (Backward Elimination)
def backward_elimination(X, y, sl=0.05):
    features = X.columns.tolist()
    while len(features) > 0:
        X_curr = X[features]
```

```
        if X_curr.empty: break
        model = sm.OLS(y, X_curr).fit()
        p_values = model.pvalues
        max_p = p_values.max()
        if max_p > sl:
            worst = p_values.idxmax()
            if worst == 'const' and len(features) > 1:
                # Try to find next worst if const is worst
                worst = p_values.drop('const').idxmax()
                if p_values[worst] <= sl: break
            features.remove(worst)
        else:
            break
    return model, features

print(f"Running Stepwise Regression on {len(X_sm.columns)} features...")
final_sm_model, selected_features = backward_elimination(X_sm, y_sm)

# 3. OUTPUTS
print(final_sm_model.summary())

# 4. METRICS TABLE
metrics_df = pd.DataFrame({
    "Metric": ["Adjusted R-Squared", "AIC", "BIC", "MSE"],
    "Value": [final_sm_model.rsquared_adj, final_sm_model.aic, final_sm_model.
 ↪bic, final_sm_model.mse_resid]
})
display(metrics_df)

# 5. REQUIRED PLOT: Train vs Test vs Predicted
plt.figure(figsize=(10, 6))
plot_n = 50
viz_df = df_stats.sample(n=plot_n*2, random_state=99).copy()
viz_encoded = pd.get_dummies(viz_df, columns=cat_cols, drop_first=True,␣
 ↪dtype=int)
viz_encoded = sm.add_constant(viz_encoded).astype(float)
viz_encoded = viz_encoded.reindex(columns=selected_features, fill_value=0)

viz_train = viz_encoded.iloc[:plot_n]
viz_test = viz_encoded.iloc[plot_n:]
viz_y_train = y_sm.loc[viz_train.index]
viz_y_test = y_sm.loc[viz_test.index]
viz_pred = final_sm_model.predict(viz_test)

plt.scatter(viz_train['Dep_Delay'], viz_y_train, color='black', alpha=0.6,␣
 ↪label='Train')
```

```
plt.scatter(viz_test['Dep_Delay'], viz_y_test, color='red', alpha=0.6,␣
 ↪label='Test')
plt.scatter(viz_test['Dep_Delay'], viz_pred, color='blue', marker='x', s=60,␣
 ↪label='Predicted')
plt.title('Regression Analysis: Train vs Test vs Predicted')
plt.xlabel('Departure Delay'); plt.ylabel('Arrival Delay')
plt.legend(); plt.grid(True, alpha=0.3)
plt.show()
```

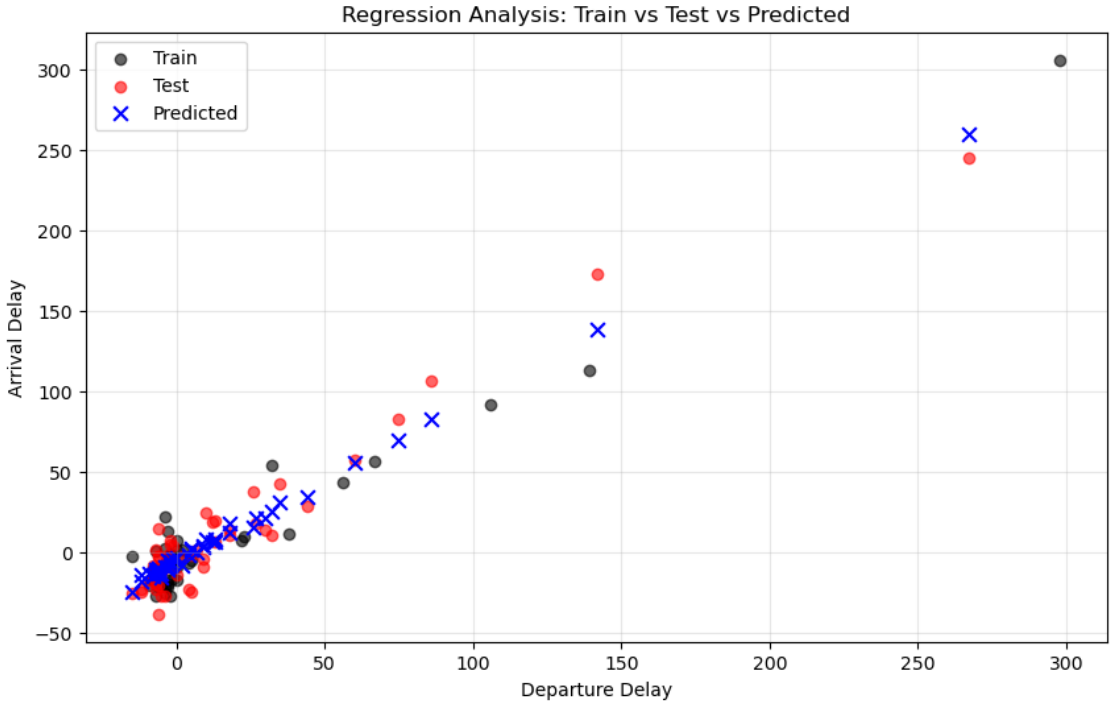
```
--- Phase II: Statistical Analysis (Syllabus Requirements) ---
Running Stepwise Regression on 78 features…
                            OLS Regression Results
==============================================================================
Dep. Variable:              Arr_Delay   R-squared:                       0.951
Model:                            OLS   Adj. R-squared:                  0.951
Method:                 Least Squares   F-statistic:                 2.710e+04
Date:                Fri, 05 Dec 2025   Prob (F-statistic):               0.00
Time:                        18:01:54   Log-Likelihood:            -4.0121e+05
No. Observations:              100000   AIC:                         8.026e+05
Df Residuals:                  99928   BIC:                         8.032e+05
Df Model:                         71
Covariance Type:            nonrobust
==============================================================================
                                                 coef    std err          t      P>|t|
[0.025      0.975]
------------------------------------------------------------------------------
const                                        -5.8917      0.586    -10.061      0.000
-7.039      -4.744
Dep_Delay                                     1.0038      0.001   1367.092      0.000
1.002       1.005
Flight_Duration                               0.0223      0.001     34.017      0.000
0.021       0.024
Aicraft_age                                   0.0725      0.006     12.252      0.000
0.061       0.084
tavg                                          0.0235      0.005      4.368      0.000
0.013       0.034
prcp                                          0.0806      0.006     14.588      0.000
0.070       0.091
wspd                                          0.0279      0.008      3.426      0.001
0.012       0.044
DepTime_label_Evening                        -0.8622      0.114     -7.548      0.000
-1.086      -0.638
DepTime_label_Morning                        -0.8458      0.099     -8.528      0.000
-1.040      -0.651
DepTime_label_Night                          -2.2946      0.254     -9.039      0.000
```

```
-2.792      -1.797
Airline_Allegiant Air                    4.2173      0.339     12.456      0.000
3.554       4.881
Airline_American Eagle Airlines Inc.     2.9108      0.258     11.262      0.000
2.404       3.417
Airline_Delta Air Lines Inc             -2.5763      0.153    -16.816      0.000
-2.877      -2.276
Airline_Frontier Airlines Inc.           2.0101      0.287      7.000      0.000
1.447       2.573
Airline_Hawaiian Airlines Inc.           6.7463      0.519     13.000      0.000
5.729       7.763
Airline_PSA Airlines                     2.2957      0.279      8.218      0.000
1.748       2.843
Airline_Republic Airways                 1.4371      0.238      6.033      0.000
0.970       1.904
Airline_Skywest Airlines Inc.            2.8352      0.177     15.998      0.000
2.488       3.183
Airline_Southwest Airlines Co.           0.5252      0.138      3.797      0.000
0.254       0.796
Airline_Spirit Air Lines                 0.9996      0.239      4.185      0.000
0.531       1.468
Airline_United Air Lines Inc.           -1.0995      0.167     -6.601      0.000
-1.426      -0.773
Origin_State_AL                         -3.6798      0.851     -4.325      0.000
-5.347      -2.012
Origin_State_AR                         -4.0970      0.888     -4.614      0.000
-5.837      -2.357
Origin_State_AS                        -16.9533      7.764     -2.184      0.029
-32.170      -1.736
Origin_State_AZ                         -4.9882      0.619     -8.063      0.000
-6.201      -3.776
Origin_State_CA                         -5.0918      0.579     -8.789      0.000
-6.227      -3.956
Origin_State_CO                         -4.2079      0.598     -7.037      0.000
-5.380      -3.036
Origin_State_CT                         -3.9669      0.935     -4.242      0.000
-5.800      -2.134
Origin_State_FL                         -4.3085      0.586     -7.349      0.000
-5.458      -3.159
Origin_State_GA                         -2.6501      0.597     -4.439      0.000
-3.820      -1.480
Origin_State_HI                         -6.1300      0.701     -8.741      0.000
-7.505      -4.755
Origin_State_IA                         -2.8536      0.925     -3.085      0.002
-4.667      -1.040
Origin_State_ID                         -3.3470      0.872     -3.840      0.000
-5.055      -1.639
Origin_State_IL                         -4.6765      0.593     -7.881      0.000
```

-5.839	-3.513				
Origin_State_IN		-5.6505	0.735	-7.685	0.000
-7.092	-4.209				
Origin_State_KS		-3.2238	1.114	-2.895	0.004
-5.407	-1.041				
Origin_State_KY		-4.0871	0.699	-5.847	0.000
-5.457	-2.717				
Origin_State_LA		-5.1235	0.714	-7.176	0.000
-6.523	-3.724				
Origin_State_MA		-6.4139	0.632	-10.149	0.000
-7.653	-5.175				
Origin_State_MD		-5.3955	0.670	-8.052	0.000
-6.709	-4.082				
Origin_State_ME		-3.3625	1.012	-3.322	0.001
-5.346	-1.379				
Origin_State_MI		-5.8031	0.626	-9.271	0.000
-7.030	-4.576				
Origin_State_MN		-5.7251	0.646	-8.857	0.000
-6.992	-4.458				
Origin_State_MO		-3.6043	0.647	-5.571	0.000
-4.872	-2.336				
Origin_State_MS		-3.7463	1.148	-3.264	0.001
-5.996	-1.497				
Origin_State_MT		-2.9490	0.894	-3.300	0.001
-4.700	-1.198				
Origin_State_NC		-2.8982	0.599	-4.837	0.000
-4.072	-1.724				
Origin_State_ND		-6.0061	1.063	-5.648	0.000
-8.090	-3.922				
Origin_State_NE		-4.6573	0.911	-5.113	0.000
-6.443	-2.872				
Origin_State_NH		-6.0985	1.436	-4.246	0.000
-8.914	-3.283				
Origin_State_NJ		-7.4730	0.639	-11.691	0.000
-8.726	-6.220				
Origin_State_NM		-2.3103	0.892	-2.589	0.010
-4.059	-0.561				
Origin_State_NV		-3.5132	0.618	-5.688	0.000
-4.724	-2.303				
Origin_State_NY		-8.2018	0.588	-13.941	0.000
-9.355	-7.049				
Origin_State_OH		-4.3317	0.665	-6.510	0.000
-5.636	-3.028				
Origin_State_OK		-3.6591	0.800	-4.576	0.000
-5.227	-2.092				
Origin_State_OR		-2.6565	0.689	-3.855	0.000
-4.007	-1.306				
Origin_State_PA		-5.0021	0.634	-7.890	0.000

```
-6.245      -3.760
Origin_State_PR                    -5.3755      0.814     -6.606      0.000
-6.970      -3.781
Origin_State_RI                    -4.6078      1.077     -4.279      0.000
-6.718      -2.497
Origin_State_SC                    -3.7233      0.726     -5.126      0.000
-5.147      -2.300
Origin_State_SD                    -4.4335      1.209     -3.668      0.000
-6.803      -2.064
Origin_State_TN                    -3.8636      0.635     -6.088      0.000
-5.107      -2.620
Origin_State_TX                    -4.1776      0.582     -7.175      0.000
-5.319      -3.036
Origin_State_UT                    -3.3802      0.653     -5.175      0.000
-4.661      -2.100
Origin_State_VA                    -5.9659      0.609     -9.801      0.000
-7.159      -4.773
Origin_State_VI                    -3.7989      1.545     -2.459      0.014
-6.827      -0.771
Origin_State_VT                    -5.2156      1.383     -3.770      0.000
-7.927      -2.504
Origin_State_WA                    -3.0778      0.616     -5.000      0.000
-4.284      -1.871
Origin_State_WI                    -4.9859      0.757     -6.590      0.000
-6.469      -3.503
Origin_State_WV                    -5.4327      1.793     -3.030      0.002
-8.947      -1.918
Origin_State_WY                    -3.0673      1.250     -2.453      0.014
-5.518      -0.617
==============================================================================
Omnibus:                    34575.277   Durbin-Watson:                   2.000
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           304305.252
Skew:                           1.416   Prob(JB):                         0.00
Kurtosis:                      11.063   Cond. No.                     2.94e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.94e+04. This might indicate that there are
strong multicollinearity or other numerical problems.
```

	Metric	Value
0	Adjusted R-Squared	0.950599
1	AIC	802554.605750
2	BIC	803239.536383
3	MSE	178.921874


Regression Analysis: Train vs Test vs Predicted
Train
Test
Predicted
Arrival Delay
Departure Delay
300
250
200
150
100
50
0
−50
0
50
100
150
200
250
300