In [16]:
```
!pip install pyswarms
```

```
Collecting pyswarms
  Downloading pyswarms-1.3.0-py2.py3-none-any.whl.metadata (33 kB)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packag
es (from pyswarms) (1.13.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packag
es (from pyswarms) (1.26.4)
Requirement already satisfied: matplotlib>=1.3.1 in /usr/local/lib/python3.1
0/dist-packages (from pyswarms) (3.7.5)
Requirement already satisfied: attrs in /usr/local/lib/python3.10/dist-packag
es (from pyswarms) (24.3.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-package
s (from pyswarms) (4.67.1)
Requirement already satisfied: future in /usr/local/lib/python3.10/dist-packa
ges (from pyswarms) (1.0.0)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.10/dist-packa
ges (from pyswarms) (6.0.2)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/
dist-packages (from matplotlib>=1.3.1->pyswarms) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist
-packages (from matplotlib>=1.3.1->pyswarms) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib>=1.3.1->pyswarms) (4.55.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib>=1.3.1->pyswarms) (1.4.7)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/d
ist-packages (from matplotlib>=1.3.1->pyswarms) (24.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib>=1.3.1->pyswarms) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/
dist-packages (from matplotlib>=1.3.1->pyswarms) (3.2.0)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python
3.10/dist-packages (from matplotlib>=1.3.1->pyswarms) (2.8.2)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-pack
ages (from numpy->pyswarms) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-p
ackages (from numpy->pyswarms) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-pa
ckages (from numpy->pyswarms) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.10/dist-packages
(from numpy->pyswarms) (2025.0.1)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.10/dist-packa
ges (from numpy->pyswarms) (2022.0.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.10/dist-
packages (from numpy->pyswarms) (2.4.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-pac
kages (from python-dateutil>=2.7->matplotlib>=1.3.1->pyswarms) (1.17.0)
Requirement already satisfied: intel-openmp>=2024 in /usr/local/lib/python3.1
0/dist-packages (from mkl->numpy->pyswarms) (2024.2.0)
Requirement already satisfied: tbb==2022.* in /usr/local/lib/python3.10/dist-
packages (from mkl->numpy->pyswarms) (2022.0.0)
Requirement already satisfied: tcmlib==1.* in /usr/local/lib/python3.10/dist-
packages (from tbb==2022.*->mkl->numpy->pyswarms) (1.2.0)
Requirement already satisfied: intel-cmplr-lib-rt in /usr/local/lib/python3.1
0/dist-packages (from mkl_umath->numpy->pyswarms) (2024.2.0)
Requirement already satisfied: intel-cmplr-lib-ur==2024.2.0 in /usr/local/li
b/python3.10/dist-packages (from intel-openmp>=2024->mkl->numpy->pyswarms) (2
024.2.0)
```

```
Downloading pyswarms-1.3.0-py2.py3-none-any.whl (104 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 104.1/104.1 kB 4.1 MB/s eta 0:00:
00
Installing collected packages: pyswarms
Successfully installed pyswarms-1.3.0
```

In [17]:
```
pip install tensorflow keras-tuner scikit-learn pandas numpy matplotlib
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-p
ackages (2.17.1)
Requirement already satisfied: keras-tuner in /usr/local/lib/python3.10/dist-
packages (1.4.7)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist
-packages (1.2.2)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packa
ges (2.2.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packag
es (1.26.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-p
ackages (3.7.5)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/di
st-packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.1
0/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in /usr/local/lib/python
3.10/dist-packages (from tensorflow) (24.3.25)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in /usr/lo
cal/lib/python3.10/dist-packages (from tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.
10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=3.10.0 in /usr/local/lib/python3.10/dist
-packages (from tensorflow) (3.12.1)
Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.10/
dist-packages (from tensorflow) (18.1.1)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.3.1 in /usr/local/lib/pyth
on3.10/dist-packages (from tensorflow) (0.4.1)
Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.1
0/dist-packages (from tensorflow) (3.4.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-pa
ckages (from tensorflow) (24.2)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=
4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3 in /usr/local/lib/python3.10/dist-packages
(from tensorflow) (3.20.3)
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.
10/dist-packages (from tensorflow) (2.32.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-p
ackages (from tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.17.0)
Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.10/
dist-packages (from tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/pyt
hon3.10/dist-packages (from tensorflow) (4.12.2)
Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.10/dis
t-packages (from tensorflow) (1.17.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.
10/dist-packages (from tensorflow) (1.68.1)
Requirement already satisfied: tensorboard<2.18,>=2.17 in /usr/local/lib/pyth
on3.10/dist-packages (from tensorflow) (2.17.1)
Requirement already satisfied: keras>=3.2.0 in /usr/local/lib/python3.10/dist
-packages (from tensorflow) (3.5.0)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/l
ocal/lib/python3.10/dist-packages (from tensorflow) (0.37.1)
Requirement already satisfied: kt-legacy in /usr/local/lib/python3.10/dist-pa
ckages (from keras-tuner) (1.0.5)
```

```
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist
-packages (from scikit-learn) (1.13.1)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dis
t-packages (from scikit-learn) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python
3.10/dist-packages (from scikit-learn) (3.5.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/pytho
n3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist
-packages (from pandas) (2024.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.10/di
st-packages (from pandas) (2024.2)
Requirement already satisfied: mkl_fft in /usr/local/lib/python3.10/dist-pack
ages (from numpy) (1.3.8)
Requirement already satisfied: mkl_random in /usr/local/lib/python3.10/dist-p
ackages (from numpy) (1.2.4)
Requirement already satisfied: mkl_umath in /usr/local/lib/python3.10/dist-pa
ckages (from numpy) (0.1.1)
Requirement already satisfied: mkl in /usr/local/lib/python3.10/dist-packages
(from numpy) (2025.0.1)
Requirement already satisfied: tbb4py in /usr/local/lib/python3.10/dist-packa
ges (from numpy) (2022.0.0)
Requirement already satisfied: mkl-service in /usr/local/lib/python3.10/dist-
packages (from numpy) (2.4.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/
dist-packages (from matplotlib) (1.3.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist
-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib) (4.55.3)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.1
0/dist-packages (from matplotlib) (1.4.7)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dis
t-packages (from matplotlib) (11.0.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/
dist-packages (from matplotlib) (3.2.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.1
0/dist-packages (from astunparse>=1.6.0->tensorflow) (0.45.1)
Requirement already satisfied: rich in /usr/local/lib/python3.10/dist-package
s (from keras>=3.2.0->tensorflow) (13.9.4)
Requirement already satisfied: namex in /usr/local/lib/python3.10/dist-packag
es (from keras>=3.2.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in /usr/local/lib/python3.10/dist-packa
ges (from keras>=3.2.0->tensorflow) (0.13.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/pyt
hon3.10/dist-packages (from requests<3,>=2.21.0->tensorflow) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist
-packages (from requests<3,>=2.21.0->tensorflow) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.1
0/dist-packages (from requests<3,>=2.21.0->tensorflow) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.1
0/dist-packages (from requests<3,>=2.21.0->tensorflow) (2024.12.14)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.10/d
ist-packages (from tensorboard<2.18,>=2.17->tensorflow) (3.7)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in /usr/
local/lib/python3.10/dist-packages (from tensorboard<2.18,>=2.17->tensorflow)
(0.7.2)
```

```
Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.10/d
ist-packages (from tensorboard<2.18,>=2.17->tensorflow) (3.1.3)
Requirement already satisfied: intel-openmp>=2024 in /usr/local/lib/python3.1
0/dist-packages (from mkl->numpy) (2024.2.0)
Requirement already satisfied: tbb==2022.* in /usr/local/lib/python3.10/dist-
packages (from mkl->numpy) (2022.0.0)
Requirement already satisfied: tcmlib==1.* in /usr/local/lib/python3.10/dist-
packages (from tbb==2022.*->mkl->numpy) (1.2.0)
Requirement already satisfied: intel-cmplr-lib-rt in /usr/local/lib/python3.1
0/dist-packages (from mkl_umath->numpy) (2024.2.0)
Requirement already satisfied: intel-cmplr-lib-ur==2024.2.0 in /usr/local/li
b/python3.10/dist-packages (from intel-openmp>=2024->mkl->numpy) (2024.2.0)
Requirement already satisfied: MarkupSafe>=2.1.1 in /usr/local/lib/python3.1
0/dist-packages (from werkzeug>=1.0.1->tensorboard<2.18,>=2.17->tensorflow)
(3.0.2)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python
3.10/dist-packages (from rich->keras>=3.2.0->tensorflow) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/pyth
on3.10/dist-packages (from rich->keras>=3.2.0->tensorflow) (2.18.0)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-p
ackages (from markdown-it-py>=2.2.0->rich->keras>=3.2.0->tensorflow) (0.1.2)
Note: you may need to restart the kernel to use updated packages.
```

In [94]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split, StratifiedKFold
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_sc
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
from sklearn.feature_selection import mutual_info_classif, SelectKBest
from imblearn.over_sampling import SMOTE
from imblearn.pipeline import Pipeline
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Bidirectional, LSTM, Dense, Dropout, BatchN
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLR
from tensorflow.keras.optimizers import Adam
from keras_tuner import RandomSearch
```

In [95]:
```
df = pd.read_csv("/kaggle/input/cloud-computing-performance-metrics/vmCloud_dat

df.head()
```

```
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:1458: Run
timeWarning: invalid value encountered in greater
  has_large_values = (abs_vals > 1e6).any()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:1459: Run
timeWarning: invalid value encountered in less
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).any
()
/usr/local/lib/python3.10/dist-packages/pandas/io/formats/format.py:1459: Run
timeWarning: invalid value encountered in greater
  has_small_values = ((abs_vals < 10 ** (-self.digits)) & (abs_vals > 0)).any
()
```

Out[95]:

	vm_id	timestamp	cpu_usage	memory_usage	network_traffic	power_consumption	nu
0	c5215826-6237-4a33-9312-72c1df909881	2023-01-25 09:10:54	54.881350	78.950861	164.775973	287.808986	
1	29690bc6-1f34-403b-b509-a1ecb1834fb8	2023-01-26 04:46:34	71.518937	29.901883	NaN	362.273569	
2	2e55abc3-5bad-46cb-b445-a577f5e9bf2a	2023-01-13 23:39:47	NaN	92.709195	203.674847	231.467903	
3	e672e32f-c134-4fbc-992b-34eb63bef6bf	2023-02-09 11:45:49	54.488318	88.100960	NaN	195.639954	
4	f38b8b50-6926-4533-be4f-89ad11624071	2023-06-14 08:27:26	42.365480	NaN	NaN	359.451537	

In [96]:
```
df.shape
```

Out[96]: (2000000, 12)

In [40]:
```
# Handle missing values
df = df.fillna(method='ffill')


df.shape
```

```
<ipython-input-40-1c3ca3487565>:2: FutureWarning: DataFrame.fillna with 'meth
od' is deprecated and will raise in a future version. Use obj.ffill() or obj.
bfill() instead.
  df = df.fillna(method='ffill')
```

Out[40]: (2000000, 12)

In [97]:
```
def clean_data(df):
    # Drop irrelevant columns
    df = df.drop(columns=['vm_id', 'timestamp'])

    # Handle missing values
    # Numerical columns: Impute with median
    num_cols = df.select_dtypes(include=np.number).columns
    num_imputer = SimpleImputer(strategy='median')
    df[num_cols] = num_imputer.fit_transform(df[num_cols])

    # Categorical columns: Impute with mode
    cat_cols = df.select_dtypes(include='object').columns.drop('task_status')
    cat_imputer = SimpleImputer(strategy='most_frequent')
    df[cat_cols] = cat_imputer.fit_transform(df[cat_cols])

    # Remove duplicates
    df = df.drop_duplicates()

    return df

cleaned_df = clean_data(df)
```

In [24]:
```
scaler = StandardScaler()
numerical_columns = cleaned_df.select_dtypes(include=[np.number]).columns
cleaned_df[numerical_columns] = scaler.fit_transform(cleaned_df[numerical_colum
cleaned_df.head()
```

Out[24]:

	cpu_usage	memory_usage	network_traffic	power_consumption	num_executed_instructions	e
0	0.168796	1.004291	-1.161974	0.261818	0.875470	
1	0.744872	-0.695900	-1.161974	0.777600	0.120372	
2	0.744872	1.481198	-1.027167	-0.128431	0.167154	
3	0.155187	1.321462	-1.027167	-0.376594	0.303342	
4	-0.264566	1.321462	-1.027167	0.758053	-0.568191	

In [98]:
```
# Target Conversion
def convert_target(df):
    df['task_status'] = np.where(df['task_status'].isin(['completed', 'finished
    return df

cleaned_df = convert_target(cleaned_df)

cleaned_df.head()
```

Out[98]:

	cpu_usage	memory_usage	network_traffic	power_consumption	num_executed_instructions	e
0	54.881350	78.950861	164.775973	287.808986	7527.0	
1	71.518937	29.901883	500.007595	362.273569	5348.0	
2	50.054758	92.709195	203.674847	231.467903	5483.0	
3	54.488318	88.100960	500.007595	195.639954	5876.0	
4	42.365480	49.976089	500.007595	359.451537	3361.0	

In [99]:
```
# Feature Selection
def select_features(X_train, y_train, X_test, k='all'):
    selector = SelectKBest(score_func=f_classif, k=k)
    X_train_selected = selector.fit_transform(X_train, y_train)
    X_test_selected = selector.transform(X_test)
    return X_train_selected, X_test_selected
```

In [44]:
```
cleaned_df = cleaned_df.drop(columns=['task_type','task_priority'])
cleaned_df.head()
```

Out[44]:

	cpu_usage	memory_usage	network_traffic	power_consumption	num_executed_instructions	e
0	54.881350	78.950861	164.775973	287.808986	7527.0	
1	71.518937	29.901883	164.775973	362.273569	5348.0	
2	71.518937	92.709195	203.674847	231.467903	5483.0	
3	54.488318	88.100960	203.674847	195.639954	5876.0	
4	42.365480	88.100960	203.674847	359.451537	3361.0	

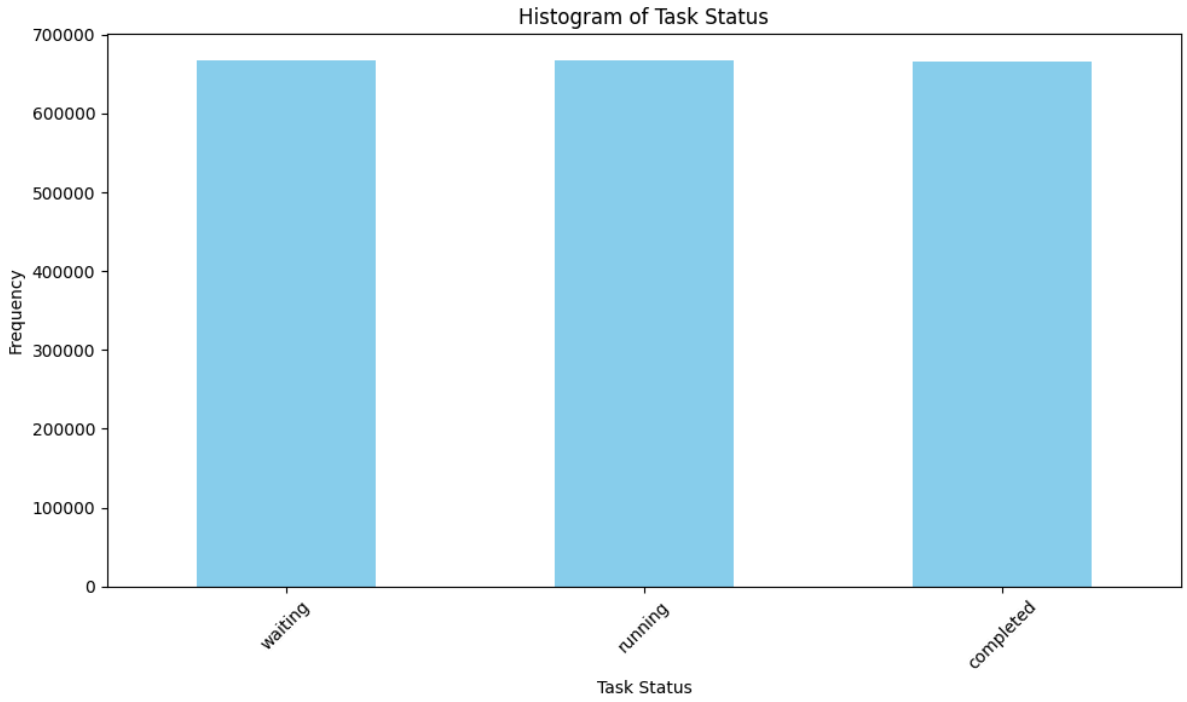
In [28]:
```
# Create a histogram for the 'task_status' column
plt.figure(figsize=(10, 6))  # Set the figure size

# Plot the histogram
df['task_status'].value_counts().plot(kind='bar', color='skyblue')

# Add title and labels
plt.title('Histogram of Task Status')
plt.xlabel('Task Status')
plt.ylabel('Frequency')

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Show the plot
plt.tight_layout()  # Adjust layout to prevent label cutoff
plt.show()
```



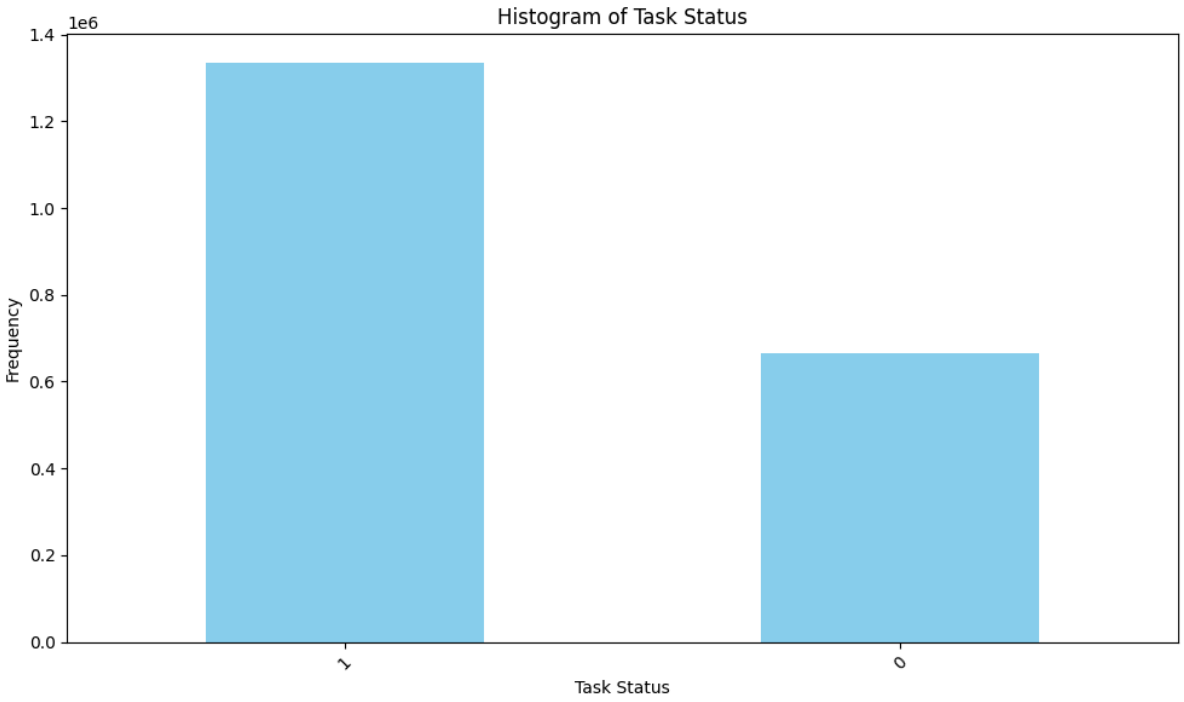
In [29]:
```
# Create a histogram for the 'task_status' column
plt.figure(figsize=(10, 6))  # Set the figure size

# Plot the histogram
cleaned_df['task_status'].value_counts().plot(kind='bar', color='skyblue')

# Add title and labels
plt.title('Histogram of Task Status')
plt.xlabel('Task Status')
plt.ylabel('Frequency')

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Show the plot
plt.tight_layout()  # Adjust layout to prevent label cutoff
plt.show()
```



In [100]:

```
# Advanced Preprocessing Pipeline
def preprocess_data(df):
    y = df['task_status']
    X = df.drop(columns=['task_status'])

    numeric_features = X.select_dtypes(include=np.number).columns
    categorical_features = X.select_dtypes(include='object').columns

    # Create stratified k-fold for better validation
    skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

    # Build processing pipeline
    pipeline = Pipeline([
        ('preprocessor', ColumnTransformer([
            ('num', StandardScaler(), numeric_features),
            ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_feature
        ])),
        ('feature_selection', SelectKBest(mutual_info_classif, k='all')),
        ('smote', SMOTE(sampling_strategy=0.8, random_state=42, k_neighbors=5))
    ])

    X_processed, y_processed = pipeline.fit_resample(X, y)

    return train_test_split(
        X_processed, y_processed,
        test_size=0.2,
        stratify=y_processed,
        random_state=42
    )

X_train, X_test, y_train, y_test = preprocess_data(cleaned_df)
```

In [102]:

```python
def create_sequences(X, y, seq_length=10):  # Set default sequence length
    X_seq, y_seq = [], []
    for i in range(len(X) - seq_length):
        X_seq.append(X[i:i + seq_length])
        y_seq.append(y[i + seq_length])  # Now works with NumPy array indexing
    return np.array(X_seq), np.array(y_seq)

# Convert sparse matrices to dense arrays first
X_train_dense = X_train.toarray() if hasattr(X_train, "toarray") else X_train
X_test_dense = X_test.toarray() if hasattr(X_test, "toarray") else X_test

# Convert y_train and y_test to NumPy arrays
y_train_array = y_train.values  # Convert to NumPy array
y_test_array = y_test.values    # Convert to NumPy array

# Create sequences with proper sequence length
seq_length = 10  # Optimal for cloud metrics temporal patterns
X_train_seq, y_train_seq = create_sequences(X_train_dense, y_train_array, seq_l
X_test_seq, y_test_seq = create_sequences(X_test_dense, y_test_array, seq_lengt

# Verify shapes
print(f"Training sequences shape: {X_train_seq.shape}")
print(f"Test sequences shape: {X_test_seq.shape}")
```

```
Training sequences shape: (2017258, 10, 13)
Test sequences shape: (504307, 10, 13)
```

In [87]:

```
def build_model(hp):
    model = Sequential()
    model.add(Bidirectional(
        LSTM(units=hp.Int('units', min_value=64, max_value=256, step=64),
             return_sequences=True),
        input_shape=(X_train_seq.shape[1], X_train_seq.shape[2])
    ))
    model.add(BatchNormalization())
    model.add(Dropout(hp.Float('dropout', 0.2, 0.5)))

    for i in range(hp.Int('num_layers', 1, 3)):
        model.add(LSTM(units=hp.Int(f'units_{i}', min_value=32, max_value=128,
                       return_sequences=True if i < hp.Int('num_layers', 1, 3)-
        model.add(BatchNormalization())
        model.add(Dropout(hp.Float(f'dropout_{i}', 0.2, 0.5)))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(
        optimizer=Adam(hp.Choice('learning_rate', [1e-3, 5e-4, 1e-4])),
        loss='binary_crossentropy',
        metrics=[
            'accuracy',
            tf.keras.metrics.AUC(name='auc'),
            tf.keras.metrics.Precision(name='precision'),
            tf.keras.metrics.Recall(name='recall')
        ]
    )
    return model
```

In [92]:

```
# Configure tuner
tuner = RandomSearch(
    build_model,
    objective='val_auc',
    max_trials=20,
    executions_per_trial=2,
    directory='lstm_tuning',
    project_name='cloud_perf'
)

# Enhanced callbacks
callbacks = [
    EarlyStopping(patience=15, restore_best_weights=True, monitor='val_auc'),
    ModelCheckpoint('best_model.keras', save_best_only=True),
    ReduceLROnPlateau(factor=0.5, patience=5)
]

# Hyperparameter search
tuner.search(X_train_seq, y_train_seq,
             epochs=10,
             batch_size=256,
             validation_split=0.2,
             callbacks=callbacks,
             verbose=1)

# Get best model
best_model = tuner.get_best_models(num_models=1)[0]

# Final training
history = best_model.fit(
    X_train_seq, y_train_seq,
    epochs=10,
    batch_size=256,
    validation_split=0.2,
    callbacks=callbacks,
    verbose=1
)
```

```
Reloading Tuner from lstm_tuning/cloud_perf/tuner0.json

Search: Running Trial #3

Value             |Best Value So Far |Hyperparameter
224               |224               |lstm_units
2                 |2                 |num_heads
32                |32                |key_dim
0.20937           |0.20937           |dropout
0.0001            |0.0001            |learning_rate

Epoch 1/10
```

```
Traceback (most recent call last):
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_tuner.py", line 274, in _try_run_and_update_trial
    self._run_and_update_trial(trial, *fit_args, **fit_kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_tuner.py", line 239, in _run_and_update_trial
    results = self.run_trial(trial, *fit_args, **fit_kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/tuner.py", line 314, in run_trial
    obj_value = self._build_and_fit_model(trial, *args, **copied_kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/tuner.py", line 233, in _build_and_fit_model
    results = self.hypermodel.fit(hp, model, *args, **kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/hypermodel.py", line 149, in fit
    return model.fit(*args, **kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras/src/utils/traceback_utils.py", line 122, in error_handler
    raise e.with_traceback(filtered_tb) from None
  File "/usr/local/lib/python3.10/dist-packages/keras/src/backend/tensorflow/nn.py", line 694, in binary_crossentropy
    raise ValueError(
ValueError: Arguments `target` and `output` must have the same rank (ndim). Received: target.shape=(None,), output.shape=(None, 10, 1)
```

```
---------------------------------------------------------------------------
RuntimeError                              Traceback (most recent call last)
<ipython-input-92-cfbdee3a0747> in <cell line: 19>()
     17
     18 # Hyperparameter search
---> 19 tuner.search(X_train_seq, y_train_seq,
     20              epochs=10,
     21              batch_size=256,

/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_tuner.py
in search(self, *fit_args, **fit_kwargs)
    233             self.on_trial_begin(trial)
    234             self._try_run_and_update_trial(trial, *fit_args, **fit_kw
args)
--> 235             self.on_trial_end(trial)
    236         self.on_search_end()
    237

/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_tuner.py
in on_trial_end(self, trial)
    337             trial: A `Trial` instance.
    338         """
--> 339         self.oracle.end_trial(trial)
    340         self.save()
    341

/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/oracle.py in w
rapped_func(*args, **kwargs)
    106             LOCKS[oracle].acquire()
    107             THREADS[oracle] = thread_name
--> 108         ret_val = func(*args, **kwargs)
    109         if need_acquire:
    110             THREADS[oracle] = None

/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/oracle.py in e
nd_trial(self, trial)
    586         if not self._retry(trial):
    587             self.end_order.append(trial.trial_id)
--> 588             self._check_consecutive_failures()
    589
    590         self._save_trial(trial)

/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/oracle.py in _
check_consecutive_failures(self)
    543                 consecutive_failures = 0
    544             if consecutive_failures == self.max_consecutive_failed_tr
ials:
--> 545                 raise RuntimeError(
    546                     "Number of consecutive failures exceeded the limi
t "
    547                     f"of {self.max_consecutive_failed_trials}.\n"

RuntimeError: Number of consecutive failures exceeded the limit of 3.
Traceback (most recent call last):
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_t
uner.py", line 274, in _try_run_and_update_trial
    self._run_and_update_trial(trial, *fit_args, **fit_kwargs)
```

```
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/base_tuner.py", line 239, in _run_and_update_trial
    results = self.run_trial(trial, *fit_args, **fit_kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/tuner.py", line 314, in run_trial
    obj_value = self._build_and_fit_model(trial, *args, **copied_kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/tuner.py", line 233, in _build_and_fit_model
    results = self.hypermodel.fit(hp, model, *args, **kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras_tuner/src/engine/hypermodel.py", line 149, in fit
    return model.fit(*args, **kwargs)
  File "/usr/local/lib/python3.10/dist-packages/keras/src/utils/traceback_utils.py", line 122, in error_handler
    raise e.with_traceback(filtered_tb) from None
  File "/usr/local/lib/python3.10/dist-packages/keras/src/backend/tensorflow/nn.py", line 694, in binary_crossentropy
    raise ValueError(
ValueError: Arguments `target` and `output` must have the same rank (ndim). Received: target.shape=(None,), output.shape=(None, 10, 1)
```

In [ ]:

```
from tensorflow.keras.layers import Input, LSTM, Dense, Dropout, BatchNormaliza
from tensorflow.keras.models import Model
from keras_tuner import RandomSearch
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, ReduceLR

def build_model(hp):
    input_shape = (X_train_seq.shape[1], X_train_seq.shape[2])  # (timesteps, f
    inputs = Input(shape=input_shape)

    # LSTM layer with hyperparameters
    x = LSTM(
        hp.Int('lstm_units', min_value=64, max_value=256, step=32),
        return_sequences=True,  # Keep return_sequences=True for attention
        kernel_regularizer=tf.keras.regularizers.l2(0.001)
    )(inputs)

    # MultiHeadAttention layer
    attention = MultiHeadAttention(
        num_heads=hp.Int('num_heads', min_value=2, max_value=4, step=2),
        key_dim=hp.Int('key_dim', min_value=32, max_value=64, step=32)
    )(x, x)  # Self-attention

    # Collapse time dimension using GlobalAveragePooling1D
    x = GlobalAveragePooling1D()(attention)  # Output shape: (None, units)

    # Add dense layer after pooling
    x = Dense(64, activation='relu')(x)
    x = BatchNormalization()(x)
    x = Dropout(hp.Float('dropout', 0.2, 0.5))(x)

    # Output layer
    outputs = Dense(1, activation='sigmoid')(x)

    # Build and compile the model
    model = Model(inputs=inputs, outputs=outputs)
    model.compile(
        optimizer=tf.keras.optimizers.Adam(
            hp.Choice('learning_rate', values=[1e-4, 1e-3, 1e-2])
        ),
        loss='binary_crossentropy',
        metrics=['accuracy', tf.keras.metrics.AUC(name='auc')]
    )

    return model

# Configure tuner
tuner = RandomSearch(
    build_model,
    objective='val_auc',
    max_trials=20,
    executions_per_trial=2,
    directory='lstm_tuning',
    project_name='cloud_perf'
)

# Enhanced callbacks
callbacks = [
```

```
    EarlyStopping(
        patience=15,
        restore_best_weights=True,
        monitor='val_auc',
        mode='max'
    ),
    ModelCheckpoint(
        'best_model.keras',
        save_best_only=True,
        save_weights_only=False
    ),
    ReduceLROnPlateau(
        factor=0.5,
        patience=5,
        monitor='val_auc',
        mode='max'
    )
]

# Hyperparameter search
tuner.search(
    X_train_seq, y_train_seq,
    epochs=10,
    batch_size=256,
    validation_split=0.2,
    callbacks=callbacks,
    verbose=1
)

# Get best model
best_model = tuner.get_best_models(num_models=1)[0]

# Final training
history = best_model.fit(
    X_train_seq, y_train_seq,
    epochs=20,
    batch_size=256,
    validation_split=0.2,
    callbacks=callbacks,
    verbose=1
)
```

```
Reloading Tuner from lstm_tuning/cloud_perf/tuner0.json

Search: Running Trial #3

Value             |Best Value So Far |Hyperparameter
224               |160               |lstm_units
2                 |2                 |num_heads
32                |32                |key_dim
0.20937           |0.41944           |dropout
0.0001            |0.01              |learning_rate

Epoch 1/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 483s 76ms/step - accuracy: 0.5519 - auc: 0.499
8 - loss: 0.6949 - val_accuracy: 0.5552 - val_auc: 0.5004 - val_loss: 0.6872
- learning_rate: 1.0000e-04
Epoch 2/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 475s 75ms/step - accuracy: 0.5561 - auc: 0.500
1 - loss: 0.6869 - val_accuracy: 0.5552 - val_auc: 0.5002 - val_loss: 0.6871
- learning_rate: 1.0000e-04
Epoch 3/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 480s 76ms/step - accuracy: 0.5553 - auc: 0.500
6 - loss: 0.6871 - val_accuracy: 0.5552 - val_auc: 0.5001 - val_loss: 0.6871
- learning_rate: 1.0000e-04
Epoch 4/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 472s 75ms/step - accuracy: 0.5559 - auc: 0.500
1 - loss: 0.6869 - val_accuracy: 0.5552 - val_auc: 0.4997 - val_loss: 0.6883
- learning_rate: 1.0000e-04
Epoch 5/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 472s 75ms/step - accuracy: 0.5552 - auc: 0.499
7 - loss: 0.6871 - val_accuracy: 0.5552 - val_auc: 0.5000 - val_loss: 0.6871
- learning_rate: 1.0000e-04
Epoch 6/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 471s 75ms/step - accuracy: 0.5557 - auc: 0.499
3 - loss: 0.6870 - val_accuracy: 0.5552 - val_auc: 0.5001 - val_loss: 0.6898
- learning_rate: 1.0000e-04
Epoch 7/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 473s 75ms/step - accuracy: 0.5559 - auc: 0.500
9 - loss: 0.6869 - val_accuracy: 0.5552 - val_auc: 0.5000 - val_loss: 0.6871
- learning_rate: 5.0000e-05
Epoch 8/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 476s 76ms/step - accuracy: 0.5556 - auc: 0.499
7 - loss: 0.6870 - val_accuracy: 0.5552 - val_auc: 0.5000 - val_loss: 0.6871
- learning_rate: 5.0000e-05
Epoch 9/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 473s 75ms/step - accuracy: 0.5559 - auc: 0.499
7 - loss: 0.6869 - val_accuracy: 0.5552 - val_auc: 0.5000 - val_loss: 0.6870
- learning_rate: 5.0000e-05
Epoch 10/10
6304/6304 ━━━━━━━━━━━━━━━━━━━━ 472s 75ms/step - accuracy: 0.5554 - auc: 0.500
2 - loss: 0.6870 - val_accuracy: 0.5552 - val_auc: 0.5001 - val_loss: 0.6871
- learning_rate: 5.0000e-05
Epoch 1/10
4413/6304 ━━━━━━━━━━━━━━━━━━━━ 2:10 69ms/step - accuracy: 0.5490 - auc: 0.500
6 - loss: 0.6975
```

In [ ]:
```python
# Evaluation
y_pred_proba = best_model.predict(X_test_seq)
y_pred = (y_pred_proba > 0.5).astype(int)

print("Classification Report:")
print(classification_report(y_test_seq, y_pred))
print("\nConfusion Matrix:")
print(confusion_matrix(y_test_seq, y_pred))
print(f"\nAUC-ROC: {roc_auc_score(y_test_seq, y_pred_proba):.4f}")
```

In [ ]:
```python
# Enhanced Visualization
def plot_advanced_metrics():
    plt.figure(figsize=(18, 6))

    metrics = ['accuracy', 'loss', 'precision', 'recall', 'auc']
    for i, metric in enumerate(metrics):
        plt.subplot(2, 3, i+1)
        plt.plot(history.history[metric], label='Train')
        plt.plot(history.history[f'val_{metric}'], label='Validation')
        plt.title(f'Model {metric.capitalize()}')
        plt.ylabel(metric.capitalize())
        plt.xlabel('Epoch')
        plt.legend()

    plt.tight_layout()
    plt.show()

plot_advanced_metrics()
```