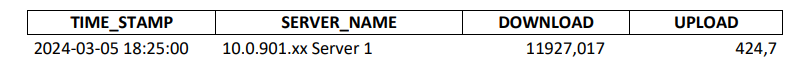
**Innova Install Future 2024 Hackathon Report – Yapay Zeka**

Soru 1: Ekte verilen veri seti üzerinden bir anomali tespit algoritması üzerinden anomali bulma çalışması yapmanız beklenmektedir. Anomaliyi tespit ederken verideki hafta içi/sonu, bayramlar, saat ve dakika kırılımına dikkat etmek gerekmektedir. Anomali olup olmadığı kontrol edilmesi beklenen veri aşağıdaki gibidir:



İlgili soruya ait kodlar innova hackathon\soru 1 dosyasında yer almaktadır .

Sorunun ilerlemesi şu şekildedir :

**Öncelikle innova hackathon\exceldeğişiklikler dosyasında yer alan excelpart.py ile elimizde yer alan orjinal excel dosyasını işlemekteyiz.**

**Nan değerlerini belirleyip index alıyoruz.**

print("Eksik değerler (indeksler):")

print(data[data.isnull().any(*axis*=1)])

#series.isna(): Series'deki eksik (NaN) değerleri belirler.İNDEX ALIYORUZ

**Arima modelini kullanarak eksik değerleri dolduruyoruz.**

*def* fill\_missing\_values(*series*):

missing\_indices = *series*[*series*.isna()].index

if not missing\_indices.empty:

model = ARIMA(*series*.dropna(), *order*=(5, 1, 0))

model\_fit = model.fit()

for idx in missing\_indices:

start = max(0, *series*.index.get\_loc(idx) - 1)

end = start + 2

forecast = model\_fit.predict(*start*=start, *end*=end)

*series*[idx] = forecast.iloc[-1]

return *series*

**#ARIMA(series.dropna(), order=(5, 1, 0)): ARIMA modelini oluşturur.**

**#5: AR (AutoRegressive) teriminin p değeri.modelin geçmiş gözlemlerine ne kadar dayalı olduğu**

**#1: I (Integrated) teriminin d değeri.ne kadar farklılaştırılacağı**

**#0: MA (Moving Average) teriminin q değeri.hata terimlerine ne kadar dayalı olduğu**

**#model.fit(): Modeli verilen eksiksiz veri üzerinde eğitir.**

data['DOWNLOAD'] = fill\_missing\_values(data['DOWNLOAD'])

data['UPLOAD'] = fill\_missing\_values(data['UPLOAD'])

**Elimizdeki verileri eğer holiday ise tatil olarak işaretliyoruz .**

**Değilse kaçıncı gün olduğuna göre label veriyoruz .**

tr\_holidays = holidays.Turkey()

*def* gun\_turu(*timestamp*):

date = *timestamp*.date()

if date in tr\_holidays:

return 'Tatil'

elif date.weekday() >= 5:

return 'Hafta Sonu'

else:

return 'Hafta İçi'

**Soru 1 de kullanmak üzere ayrı excel dosyalarına kaydediyoruz.**

hafta\_ici.reset\_index().to\_excel('exceldeğişiklikler/hafta\_ici.xlsx', *index*=False)

hafta\_sonu.reset\_index().to\_excel('exceldeğişiklikler/hafta\_sonu.xlsx', *index*=False)

tatil.reset\_index().to\_excel('exceldeğişiklikler/tatil.xlsx', *index*=False)

**Soru 1 de Isolation Forest kullanarak anomali tespiti yaptım .**

iso\_forest = IsolationForest(*n\_estimators*=100, *max\_samples*='auto', *contamination*=0.3, *max\_features*=1.0, *bootstrap*=False, *n\_jobs*=None, *random\_state*=None, *verbose*=0, *warm\_start*=False)

iso\_forest.fit(df\_scaled)

%30 oranında anomali beklenebilir

anomaly scores = iso\_forest.decision\_function(df\_scaled)

Her veri noktası için ayrı ayrı anomali skoru hesaplanır

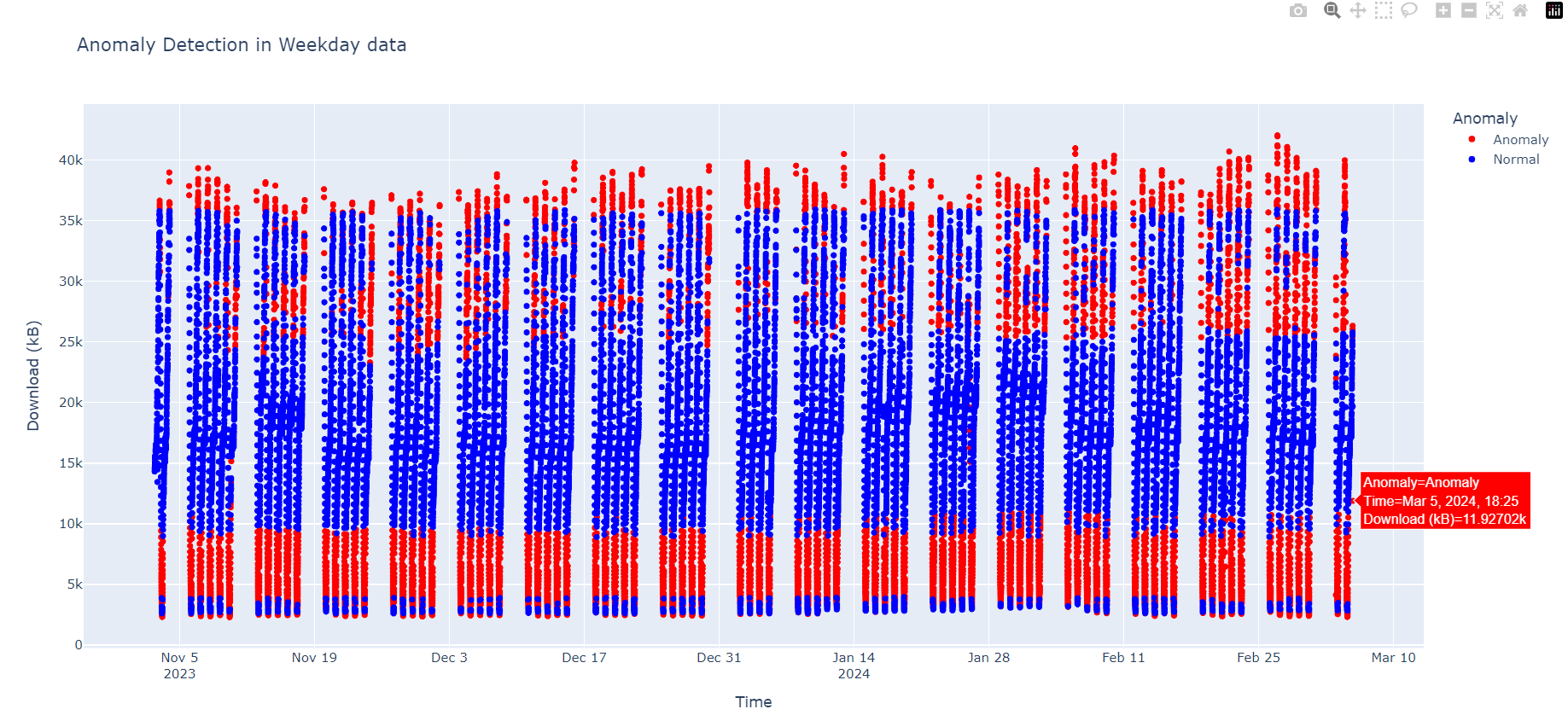
df['Anomaly'] = df['Anomaly'].apply(*lambda* *x*: 'Anomaly' if *x* == -1 else 'Normal')

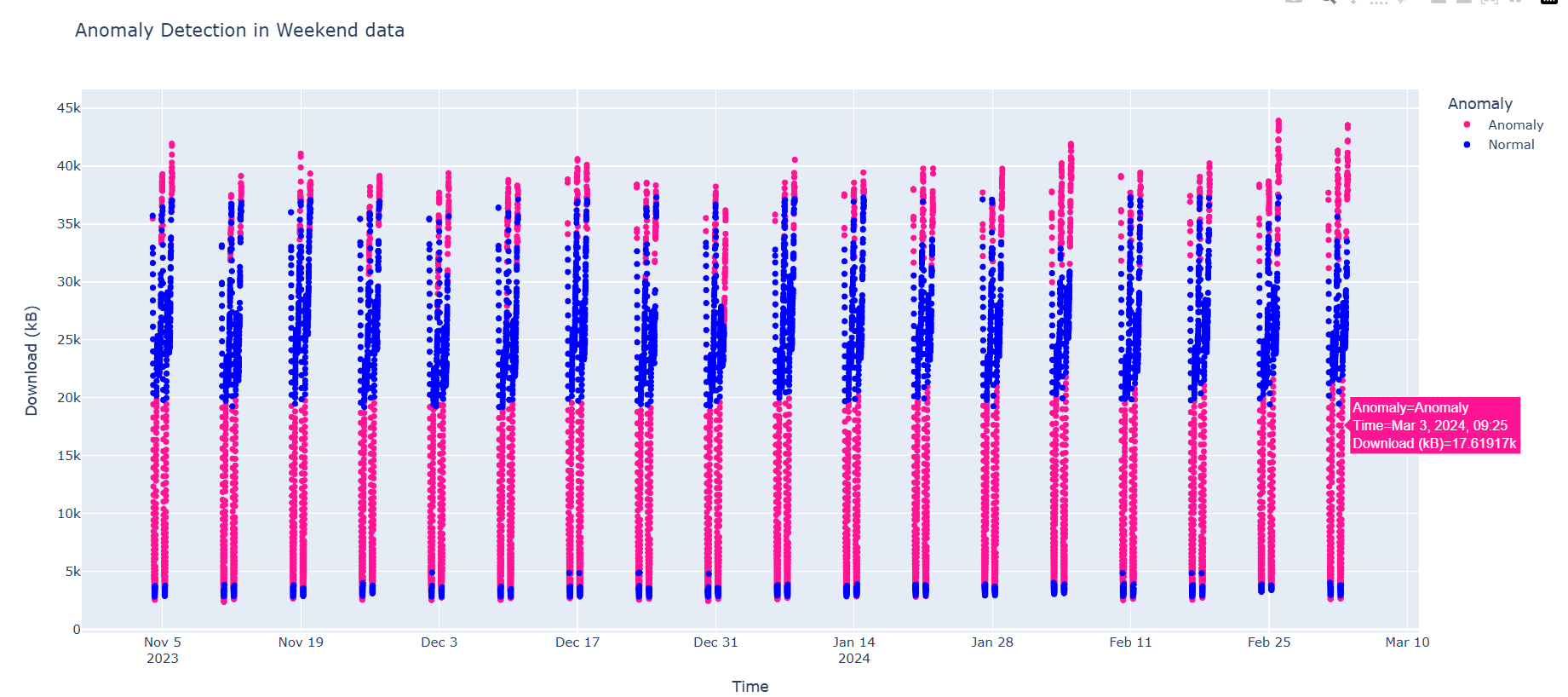
Anomali tespiti yapılır.

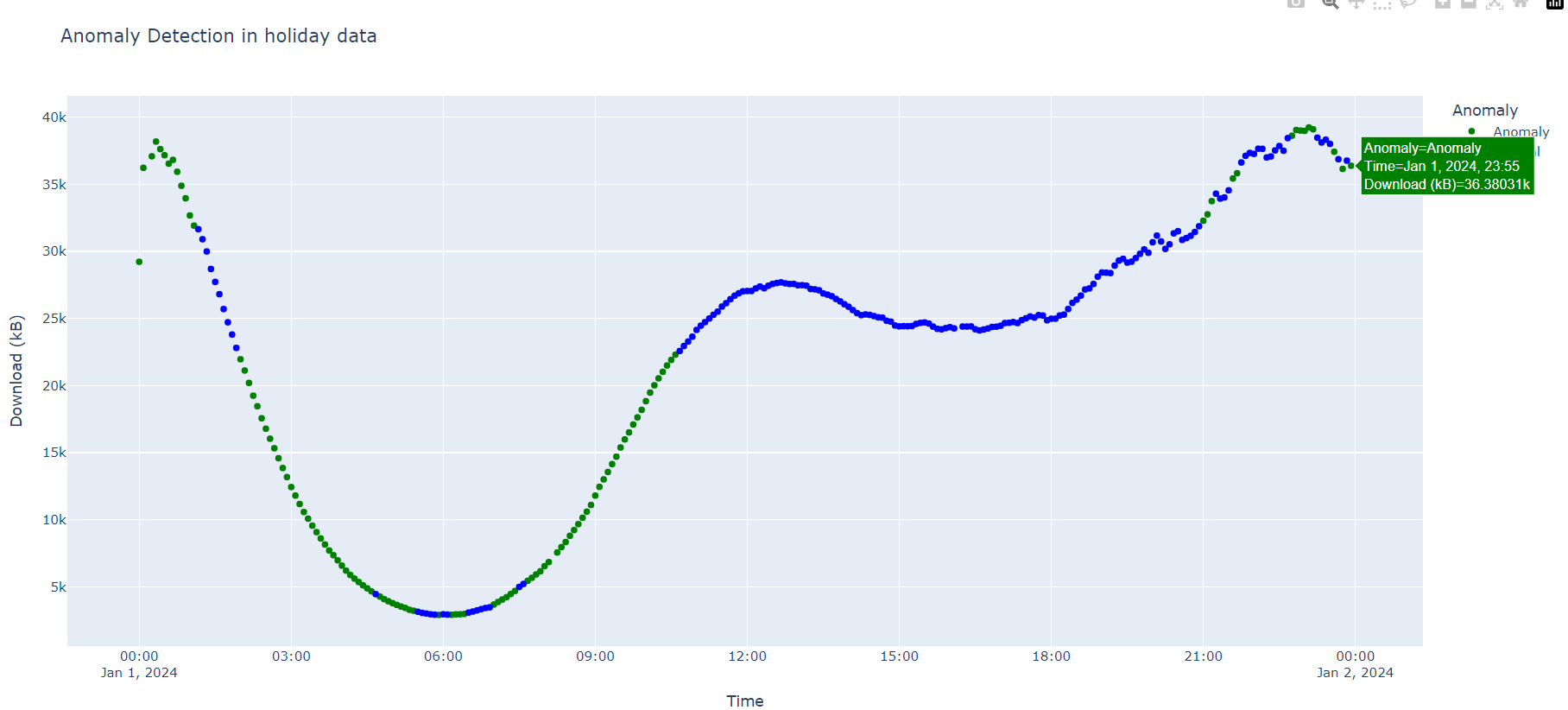
Isolation Forest her değer için ayrı anomly score hesaplar ve elimizdeki threashold değer ile bunu karşılaştırır.

Yapılan tespitin grafiği aşağıdaki gibidir ;

Hafta içi :

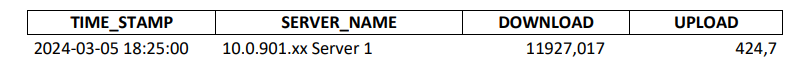
Hafta Sonu ; Tatil Günleri ;





**SORU 2**

Ekte verilen veri seti üzerinden zaman serisi tahminleme (time series forecasting) üzerinden anomali bulma çalışması yapmanız beklenmektedir. Anomali olup olmadığı kontrol edilmesi beklenen veri aşağıdaki gibidir:



2. soruda prophet kullandım.

Öncelikle orjinal excel dosyasında missingvaluesfilling.py ile eksik değerleri doldurdum.

Sonrasında prophet kullanarak dowload ve upload için 2 ayrı model eğittim.

download\_df = df[['ds', 'DOWNLOAD']].rename(*columns*={'DOWNLOAD': 'y'})

#GÜVEN ARALIĞI VE TATİL BAYRAM GÜNLERİNE GÖRE TAHMİN

download\_model = Prophet(*interval\_width*=0.95, *daily\_seasonality*=True)

download\_model.fit(download\_df)

joblib.dump(download\_model, 'soru2/download\_model.pkl')

Prophet öncesinde bu modellere göre genel bir tahminde bulunacak daha sonrada bunu asıl veri ile karşılaştıracak .

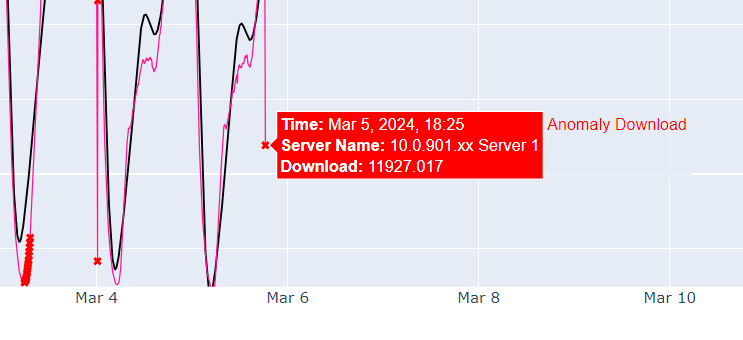
future\_download = download\_model.make\_future\_dataframe(*periods*=100, *freq*='T')

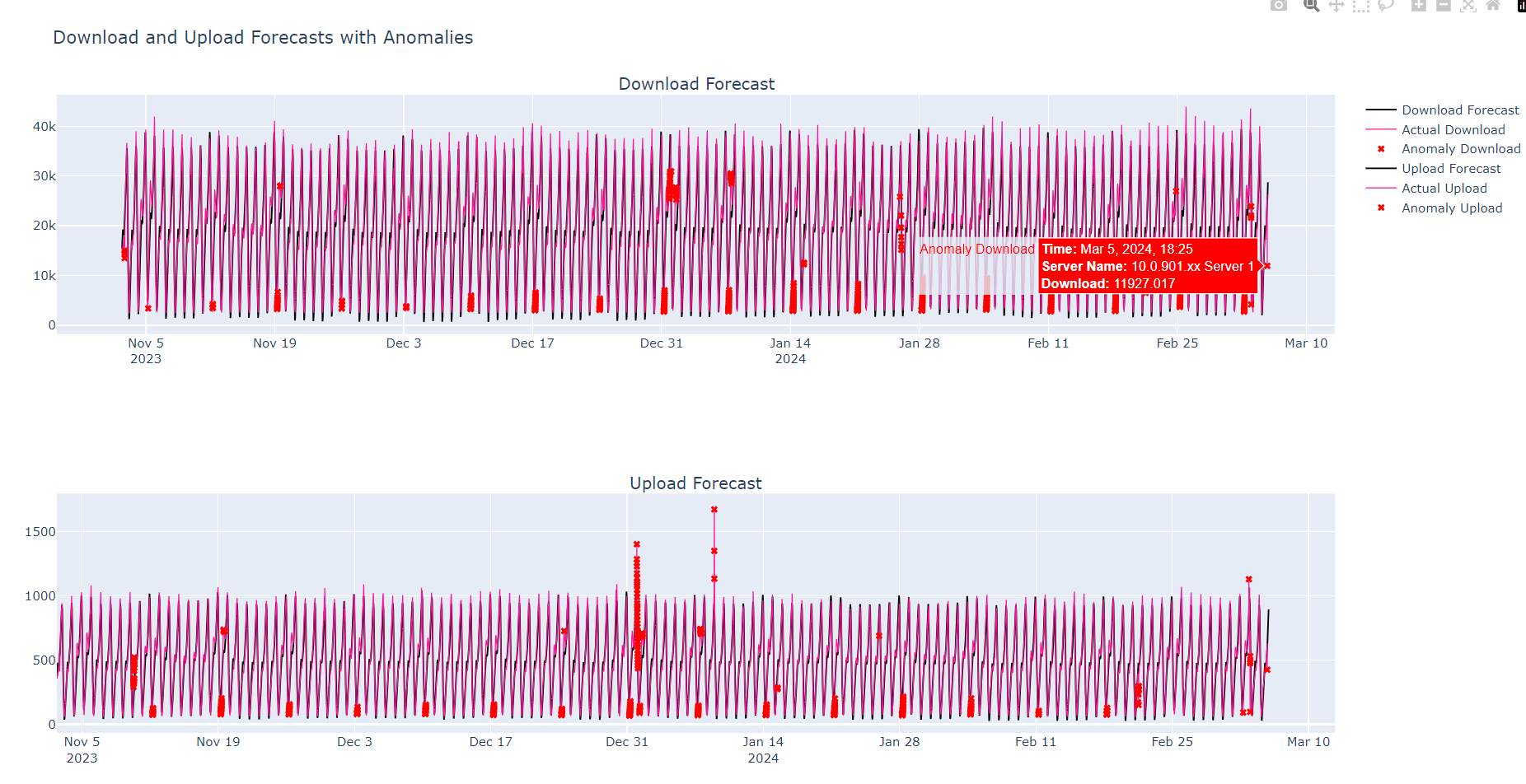
download\_merged = pd.merge(download\_df, forecast\_download[['ds', 'yhat', 'yhat\_lower', 'yhat\_upper']], *on*='ds')

download\_merged['anomaly'] = (download\_merged['DOWNLOAD'] < download\_merged['yhat\_lower']/1.1) | (download\_merged['DOWNLOAD'] > download\_merged['yhat\_upper']\*1.1)

**Eğer Dowload değeri yhatlowerdan düşükse yada yhat upper değerinden büyükse bir anomali değerdir .**

**Grafikte pembe olan çizgiler prophet tahmini siyahlar gerçek tahmindir.**





**Bonus Soru**

Mevcut sisteme yeni bir Server 2 eklenmesi planlanmaktadır. Server 1’in trafik verileri üzerinden trafiğin en yoğun olduğu saat dilimlerini hesaplayınız

Verileri saat dilimlerine göre gruplayarak, her saat diliminde 'DOWNLOAD' ve 'UPLOAD' sütunlarının toplamını hesapladım.

hourly\_traffic = df.groupby('hour')[['DOWNLOAD', 'UPLOAD']].sum().reset\_index()

hourly\_traffic['TOTAL\_TRAFFIC'] = hourly\_traffic['DOWNLOAD'] + hourly\_traffic['UPLOAD']

peak\_hours = hourly\_traffic.sort\_values(*by*='TOTAL\_TRAFFIC', *ascending*=False).head(3) # En yoğun 3 saat dilimi

peak\_hours\_text = '\n'.join([*f*"- {row['hour']}:00 - {row['hour']+1}:00 -" for index, row in peak\_hours.iterrows()])

hourly\_traffic veri çerçevesindeki saat dilimlerini toplam trafik miktarına göre sıralar .

