Combining Datasets & Data Cleaning

In [ ]:
```
import numpy as np
import pandas as pd
```

In [ ]:
```
pip install holidays
```

In [ ]:
```
from datetime import date
import holidays
```

Bike Share Toronto Data (2018)

In [ ]:
```
# Load Bike Share Toronto data
b18_q1 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/2018-Q1.cs
b18_q2 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/2018-Q2.cs
b18_q3 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/2018-Q3.cs
b18_q4 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/2018-Q4.cs
```

In [ ]:
```
# Bike Share - 2018 Q1
b18_q1.head(1)
```

In [ ]:
```
# Bike Share - 2018 Q2
b18_q2.head(1)
```

In [ ]:
```
# Bike Share - 2018 Q3
b18_q3.head(1)
```

In [ ]:
```
# Bike Share - 2018 Q4
b18_q4.head(1)
```

In [ ]:
```
# Combine quarterly datasets
toronto = [b18_q1, b18_q2, b18_q3, b18_q4]
bike_share = pd.concat(toronto).reset_index(drop=True)
bike_share.head()
```

In [ ]:
```
bike_share.tail()
```

In [ ]:
```
# Number of observations and features
bike_share.shape
```

In [ ]:
```
# Datatypes
bike_share.info()
```

In [ ]:
```
# Convert trip_start_time and trip_stop_time to DateTime Format
cols = ['trip_start_time','trip_stop_time']
bike_share[cols] = bike_share[cols].apply(pd.to_datetime)
```

In [ ]:
```
bike_share.info()
```

In [ ]:
```
# Check for duplicates
bike_share.duplicated().sum()
```

In [ ]:
```
# Check for null values
bike_share.isna().sum()
```

In [ ]:
```
pd.set_option('display.float_format', lambda x: '%.3f' % x)
bike_share.describe()
```

In [ ]:
```
# Identify weekday vs weekend
bike_share['trip_start_time'].dt.dayofweek
# Monday = 0, Sunday = 6
```

In [ ]:
```
# Create boolean weekend column
bike_share['Weekend'] = ((bike_share['trip_start_time'].dt.dayofweek) > 5).astype(int)
bike_share['Weekend'].value_counts()
# 0 = Weekday, 1 = Weekend
```

In [ ]:
```
# Identify holidays
ca_holidays = holidays.CA(years=2018)
```

In [ ]:
```
for date, occasion in ca_holidays.items():
   print(f'{date} - {occasion}')
```

In [ ]:
```
# Create boolean holiday column
bike_share['Holiday'] = pd.to_datetime(bike_share['trip_start_time']).dt.date.isin(ca_holidays).astype(int)
```

In [ ]:
```
bike_share['Holiday'].value_counts()
```

Toronto Weather Data (2018)

In [ ]:
```
# Load Toronto weather data
w1 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w2 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w3 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w4 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w5 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w6 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w7 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w8 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w9 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/e
w10 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/
w11 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/
w12 = pd.read_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/
```

In [ ]:
```
toronto_w = [w1, w2, w3, w4, w5, w6, w7, w8, w9, w10, w11, w12]
weather = pd.concat(toronto_w).reset_index(drop=True)
weather.head()
```

In [ ]:
```
# Number of observations and features
weather.shape
```

In [ ]:
```
# Datatypes
weather.info()
```

In [ ]:
```
# Classify wind as strong wind using boolean column. 1 if >= 60 km/h, 0 if < 60 km/h
weather['Strong Wind'] = (weather['Wind Spd (km/h)']>=60).astype(int)
```

In [ ]:
```
# Drop features not needed
weather.drop(['Longitude (x)',
              'Latitude (y)',
              'Station Name',
              'Climate ID',
              'Year',
              'Month',
              'Day',
              'Time',
              'Temp Flag',
              'Dew Point Temp Flag',
              'Rel Hum Flag',
              'Wind Dir (10s deg)',
              'Wind Dir Flag',
              'Wind Spd (km/h)',
              'Wind Spd Flag',
              'Visibility Flag',
              'Stn Press Flag',
              'Hmdx Flag',
              'Wind Chill Flag'], axis=1, inplace=True)
```

In [ ]:
```
weather.head()
```

In [ ]:
```
weather.Weather.unique()
```

In [ ]:
```
# Try to simplify weather feature. Transform weather into a sparse matrix
from sklearn.feature_extraction.text import CountVectorizer
```

In [ ]:
```
weather['Weather'].fillna(value='clear', inplace=True) # fill in null values with stand in value
```

In [ ]:
```
weather_code = CountVectorizer(token_pattern=r'(?u)[a-zA-Z][a-z ]+')
weather_code_f_t = weather_code.fit_transform(weather['Weather'])
print(weather_code.get_feature_names())
```

In [ ]:
```
# Instantiate into dataframe and drop stand in value
weather_code_df = pd.DataFrame(columns=weather_code.get_feature_names(),
                               data=weather_code_f_t.toarray())
weather_code_df.drop(['clear'], axis=1, inplace=True)
weather_code_df.columns = map(str.title, weather_code_df.columns)
```

In [ ]:
```
# Overwrite pervious weather with weather_code and drop weather feature
weather = pd.concat([weather, weather_code_df], join='inner', axis=1)
weather.drop(['Weather'], axis=1, inplace=True)
```

In [ ]:
```
weather.head()
```

In [ ]:
```
weather.info()
```

In [ ]:
```
weather = weather.iloc[: , :-9]
```

In [ ]:
```
weather.info()
```

In [ ]:
```
# Convert Date/Time to DateTime format
weather['Date/Time'] = pd.to_datetime(weather['Date/Time'], yearfirst=True, infer_datetime_format=True)
```

In [ ]:
```
# Check for duplicates
weather.duplicated().sum()
```

In [ ]:
```
# Check for null values
weather.isna().sum()
```

In [ ]:
```
# Handle missing values. Impute hmdx and wind chill with 0. Forwardfill any remaining null values
weather.fillna({'Hmdx': 0, 'Wind Chill': 0}, inplace=True)
weather.fillna(method='ffill', inplace=True)
```

In [ ]:
```
weather.isna().sum()
```

In [ ]:
```
weather.info()
```

In [ ]:
```
weather.head()
```

In [ ]:
```
# Save csv for EDA
weather.to_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/Weather 2018/wea
```

Combine Bike Share & Weather Data

In [ ]:
```
bike_share2 = bike_share.copy()
```

In [ ]:
```
# Format date similar to weather data
bike_share2['trip_start_time'] = bike_share2['trip_start_time'].dt.strftime('%Y-%m-%d %H:00')
```

In [ ]:
```
bike_share2.head()
```

In [ ]:
```
# Group trips by date and hour
dates = {}
for d in bike_share2.trip_start_time:
    if d not in dates:
        dates[d] = 1
    else:
        dates[d] += 1
```

In [ ]:
```
df = pd.DataFrame.from_dict(dates, orient = "index")
df['date'] = df.index
df['trips'] = df.iloc[:,0]
```

In [ ]:
```
df.head()
```

In [ ]:
```
df2 = pd.DataFrame(df.date)
df2['trips'] = df['trips']
df2.reset_index(drop = True, inplace = True)
```

In [ ]:
```
df2.head()
```

In [ ]:
```
df2['date'] = df2['date'].astype('datetime64')
```

In [ ]:
```
# Join weather data
joined = df2.join(weather.set_index('Date/Time'), on='date')
```

In [ ]:
```
joined.head()
```

In [ ]:
```
# Re-add weekend and holiday boolean columns
joined['Weekend'] = ((joined['date'].dt.dayofweek) > 5).astype(int)
joined['Holiday'] = pd.to_datetime(joined['date']).dt.date.isin(ca_holidays).astype(int)
```

In [ ]:
```
joined.head()
```

In [ ]:
```
joined.info()
```

In [ ]:
```
# Save csv for modeling
joined.to_csv(r"C:\Users\on3_a\Documents\Data Analytics, Big Data & Predictive Analytics/Data/bikeshare_weather
```