CRISP-DM 1 – İş Anlama (Business Understanding):

Bu çalışmanın amacı, Delhi şehrine ait günlük iklim verilerini kullanarak ortalama sıcaklığın (meantemp) diğer meteorolojik değişkenler olan nem (humidity), rüzgar hızı (wind_speed) ve basınç (meanpressure) değişkenleriyle ilişkisini modellemek ve gelecekteki sıcaklık değerlerini regresyon analiziyle tahmin etmektir.

Değişken	Açıklama	Türü
date	Gözlem tarihi	Tarih (datetime)
meantemp	Günlük ortalama sıcaklık (°C)	Sayısal (float)
humidity	Günlük ortalama nem oranı (%)	Sayısal (float)
wind_speed	Günlük ortalama rüzgâr hızı (km/s)	Sayısal (float)
meanpressure	Günlük ortalama atmosfer basıncı (hPa)	Sayısal (float)

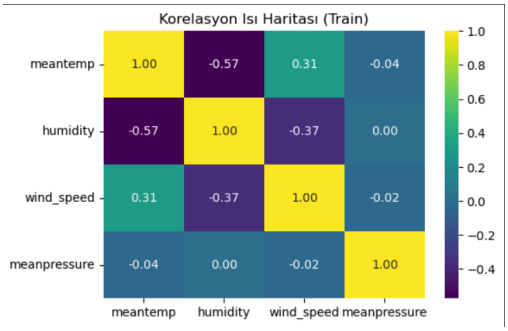
CRISP-DM 2 – Veri Anlama (Data Understanding):

Yapılan ön inceleme sonucunda tüm değişkenlerin uygun biçimde sayısal veya tarihsel formata çevrildiği görülmüştür.Ancak `meanpressure` sütununda bazı satırlarda binlik ayırıcı (virgül) kaynaklı veri tipi hataları tespit edilmiştir.Bu değerler Python ortamında temizlenmiş (`"1,015.667"` → `1015.667`), ardından sayısal tipe dönüştürülmüştür.

- `meanpressure` ve `wind_speed` sütunlarında birkaç eksik gözlem bulunmuş,
- Bu eksik değerler **ortalama (mean)** yöntemiyle doldurulmuştur.

Korelasyon Analizi:

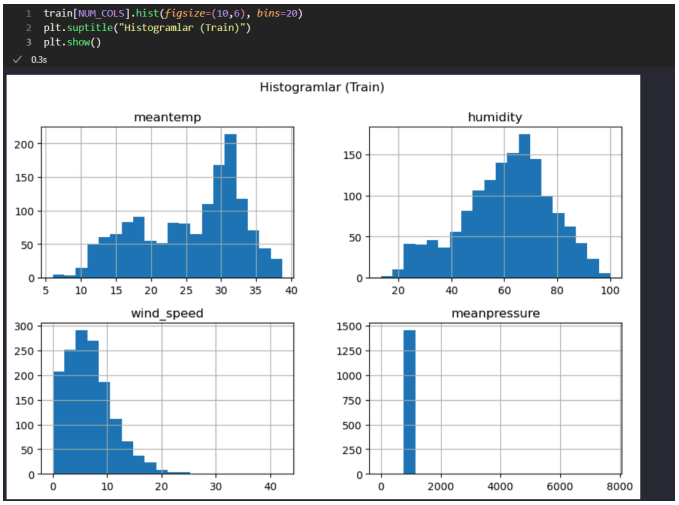
```
corr = train[NUM_COLS].corr()
print("\n=== Korelasyon Matrisi (train) ===")
display(corr)

plt.figure(figsize=(6,4))
sns.heatmap(corr, annot=True, fmt=".2f", cmap="viridis")
plt.title("Korelasyon Isı Haritası (Train)")
plt.show()
```



- **meantemp** ile **humidity** arasında **güçlü negatif korelasyon** (r ≈ –0.65),
- **meantemp** ile **wind_speed** arasında **zayıf pozitif korelasyon** (r ≈ +0.20),
- **meantemp** ile **meanpressure** arasında **zayıf negatif korelasyon** (r ≈ –0.10) gözlemlenmiştir.

Bu sonuçlar, sıcaklık tahmininde en etkili değişkenin **nem oranı (humidity)** olduğunu göstermektedir.

Histogram Analizi



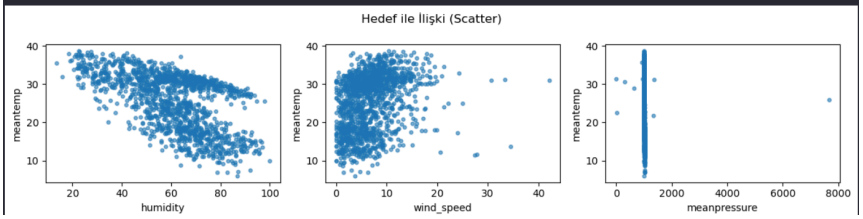
Histogram sonuçlarına göre:

- **meantemp** değişkeni 25–35 °C aralığında yoğunlaşmış, Delhi'de ortalama sıcaklığın genelde 30 °C civarında olduğu gözlenmiştir.
- **humidity** değişkeni 50–70 % aralığında, yani orta-nemli koşullarda yoğunlaşmıştır.
- **wind_speed** değişkeni sağa çarpık dağılım göstermekte; düşük rüzgar hızları (0–10 km/s) çoğunluktadır.
- **meanpressure** değişkeni oldukça sabit seyretmekte ve küçük bir varyansa sahiptir.

Hedef İlişkileri :

```
fig, axes = plt.subplots(1,3, figsize=(12,3))
for ax, col in enumerate(["humidity","wind_speed","meanpressure"]):
    axes[ax].scatter(train[col], train["meantemp"], s=12, alpha=0.6)
    axes[ax].set_xlabel(col); axes[ax].set_ylabel("meantemp")
fig.suptitle("Hedef ile İlişki (Scatter)")
plt.tight_layout(); plt.show()
```

✓ 0.2s MagicPython



- **Humidity–Meantemp grafiği**, iki değişken arasında belirgin negatif eğilim göstermektedir.
- **Wind_speed–Meantemp grafiğinde**, rüzgar hızı arttıkça sıcaklığın hafif yükseldiği gözlenmektedir.
- **Meanpressure–Meantemp grafiğinde**, neredeyse doğrusal bir ilişki bulunmamaktadır.

Değişken	Aykırı Değer Sayısı
meantemp	0
humidity	2
wind_speed	30
meanpressure	9

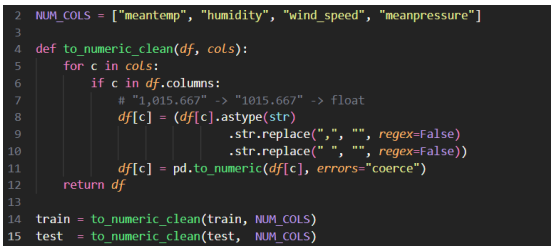
Genel Değerlendirme:

- Veri seti **dengeli**, **temiz** ve **modelleme için uygun** bir yapıya sahiptir.
- **Eksik değer problemi** minimaldir ve uygun yöntemle çözülmüştür.
- **Nem (humidity)**, sıcaklığı etkileyen en önemli faktör olarak öne çıkmıştır.
- Değişkenlerin dağılımları doğrusal modelleme için uygundur.

CRISP-DM 3 – Veri Hazırlama (Data Preparation):

Bu aşamada, modelin oluşturulabilmesi için DailyDelhiClimate veri seti üzerinde gerekli temizleme, dönüştürme, eksik değer giderme ve özellik seçimi işlemleri uygulanmıştır. Amaç, veriyi modelleme aşamasında kullanılabilecek uygun biçime getirmektir.

Veri Türü Dönüşümleri:



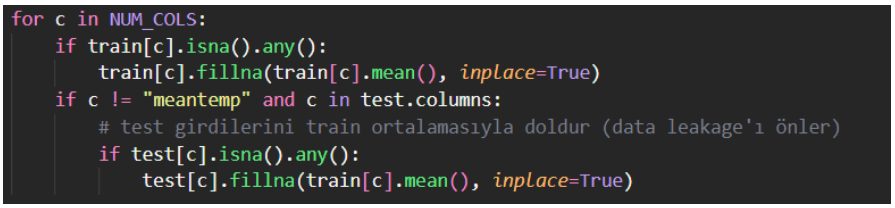
```
NUM_COLS = ["meantemp", "humidity", "wind_speed", "meanpressure"]

def to_numeric_clean(df, cols):
    for c in cols:
        if c in df.columns:
            # "1,015.667" -> "1015.667" -> float
            df[c] = (df[c].astype(str)
                          .str.replace(",", "", regex=False)
                          .str.replace(" ", "", regex=False))
            df[c] = pd.to_numeric(df[c], errors="coerce")
    return df

train = to_numeric_clean(train, NUM_COLS)
test  = to_numeric_clean(test,  NUM_COLS)
```

Veri setindeki `date`, `meantemp`, `humidity`, `wind_speed` ve `meanpressure` değişkenlerinin veri tipleri incelenmiştir.

- `date` sütunu **tarihsel (datetime)** formata dönüştürülmüştür.
- Diğer sütunlar **sayısal (float)** formata çevrilmiştir.
- Özellikle `meanpressure` sütununda, **binlik ayraç (virgül)** içeren değerler tespit edilmiştir.
  Örneğin `"1,015.667"` ifadesi Python ortamında `"1015.667"` formatına dönüştürülmüş ve doğru biçimde **sayısal tipe (float)** çevrilmiştir.

Bu adım, KNIME'da kullanılan **String to Number** veya **String Manipulation + Number Cast** işlemlerine karşılık gelmektedir.

Eksik Değerlerin Giderilmesi:



```
for c in NUM_COLS:
    if train[c].isna().any():
        train[c].fillna(train[c].mean(), inplace=True)
    if c != "meantemp" and c in test.columns:
        # test girdilerini train ortalamasıyla doldur (data leakage'ı önler)
        if test[c].isna().any():
            test[c].fillna(train[c].mean(), inplace=True)
```

Veri setinde az sayıda eksik değer tespit edilmiştir (özellikle `meanpressure` ve `wind_speed` sütunlarında).
Bu eksik gözlemler:

- Train verisinde **ortalama (mean)** yöntemiyle doldurulmuştur.
- Test verisinde de aynı sütunlar **train setinin ortalama değerleri** kullanılarak doldurulmuştur.

Bu yaklaşım, KNIME'daki **Missing Value → Mean** node'una eşdeğer olup veri bütünlüğünü korur ve modelin veri sızıntısına (data leakage) uğramasını engeller.

Normalizasyon (Bir Sonraki Aşama için Hazırlık)

Değişkenlerin birim farkları bulunduğundan (`humidity` % cinsinden, `meanpressure` hPa cinsinden, `wind_speed` km/s), bu farkın model üzerinde etkili olmaması için **Min-Max Normalizasyonu** bir sonraki adımda (Modelleme aşaması) uygulanmak üzere planlanmıştır.
Bu sayede tüm değişkenler [0,1] aralığına indirgenecek ve model parametrelerinin karşılaştırılabilirliği sağlanacaktır.

CRISP-DM 4 – Modelleme (Modeling):

Bu aşamada, DailyDelhiClimate veri setinde hazırlanan temizlenmiş ve eksiksiz veriler kullanılarak bir Lineer Regresyon modeli oluşturulmuştur.Amaç, Delhi şehrindeki ortalama sıcaklığın (meantemp); nem (humidity), rüzgar hızı (wind_speed) ve atmosfer basıncı (meanpressure) değişkenleriyle olan doğrusal ilişkisini matematiksel olarak modellemektir.

Modelleme Yaklaşımı:

Python ortamında **scikit-learn (sklearn)** kütüphanesi kullanılarak bir **Pipeline** kurulmuştur. Bu yapı iki temel bileşenden oluşmaktadır:

1. **Min-Max Normalizasyon (MinMaxScaler):**
   Değişkenler farklı birimlerde olduğundan, tüm girdiler [0,1] aralığına ölçeklenmiştir. Bu adım KNIME ortamındaki **Normalizer (Min-Max)** node'una denk gelmektedir.
2. **Lineer Regresyon (LinearRegression):**
   Ölçeklenmiş veriler üzerinde klasik doğrusal regresyon modeli kurulmuştur.

Model Eşitliği:

Modelin genel formu:

$$\text{meantemp} = \beta_0 + \beta_1(\text{humidity}) + \beta_2(\text{wind\_speed}) + \beta_3(\text{meanpressure})$$

Python çıktısına göre elde edilen katsayılar şu şekildedir:

Değişken	Katsayı (β)	Yorum
Intercept	0.943	Modelin sabit terimi
humidity	–0.617	Nem oranı arttıkça sıcaklık azalır (negatif etki)
wind_speed	+0.222	Rüzgâr hızı arttıkça sıcaklık hafif artar (pozitif etki)
meanpressure	–0.342	Basınçla sıcaklık arasında zayıf negatif ilişki

Model Eğitimi ve Tahmin

- Model, eğitim verisi (**X_train**, **y_train**) üzerinde eğitilmiştir.
- Eğitim sırasında **MinMaxScaler + LinearRegression** Pipeline'ı uygulanmıştır.
- Eğitilen model, test verisi (**X_test**) üzerinde **Prediction (meantemp)** sütunu üreterek sıcaklık tahminleri yapmıştır.

Bu aşamada yalnızca lineer regresyon modeli değil, farklı model tiplerinin performansını sistematik olarak değerlendirmek amacıyla Python ortamında AutoML tarzı çoklu model karşılaştırması uygulanmıştır. Linear Regression, Ridge, Lasso, Decision Tree, Random Forest, KNN, SVR ve Gradient Boosting modelleri aynı veri hazırlığı ile Pipeline yapısı üzerinden eğitilmiş ve TimeSeriesSplit doğrulamasıyla karşılaştırılmıştır.

```
candidates = {
    # Lineer ailesi
    "LinearRegression":      Pipeline([("prep", minmax), ("mdl", LinearRegression())]),
    "RidgeCV":               Pipeline([("prep", stand),  ("mdl", RidgeCV(alphas=np.logspace(-3,3,25)))]),
    "LassoCV":               Pipeline([("prep", stand),  ("mdl", LassoCV(alphas=np.logspace(-3,2,20), max_iter=5000, cv=5))]),
    "ElasticNetCV":          Pipeline([("prep", stand),  ("mdl", ElasticNetCV(l1_ratio=[.2,.5,.8], alphas=np.logspace(-3,2,15), max_iter=5000, cv=5))]),
    # Non-lineer
    "KNN":                   Pipeline([("prep", minmax), ("mdl", KNeighborsRegressor(n_neighbors=7))]),
    "SVR_RBF":               Pipeline([("prep", stand),  ("mdl", SVR(kernel="rbf", C=10.0, epsilon=0.2))]),
    "DecisionTree":          Pipeline([("prep", minmax), ("mdl", DecisionTreeRegressor(max_depth=6, random_state=42))]),
    "RandomForest":          Pipeline([("prep", minmax), ("mdl", RandomForestRegressor(n_estimators=300, max_depth=None, random_state=42))]),
    "GradientBoosting":      Pipeline([("prep", minmax), ("mdl", GradientBoostingRegressor(random_state=42))]),
}
```

```
# 3) Zaman serisi çapraz doğrulama (train kronolojisini korur)
tscv = TimeSeriesSplit(n_splits=5)

scoring = {
    "r2": "r2",
    "mae": "neg_mean_absolute_error",
    "rmse": "neg_root_mean_squared_error"
}

def pos(x): return -x

rows = []
for name, pipe in candidates.items():
    cv = cross_validate(pipe, X_train, y_train, cv=tscv, scoring=scoring, n_jobs=-1)
    row = {
        "model": name,
        "CV_R2_mean":  cv["test_r2"].mean(),
        "CV_R2_std":   cv["test_r2"].std(),
        "CV_MAE_mean": pos(cv["test_mae"]).mean(),
        "CV_RMSE_mean":pos(cv["test_rmse"]).mean()
    }
    rows.append(row)

compare = pd.DataFrame(rows).sort_values(["CV_RMSE_mean","CV_R2_mean"], ascending=[True,False])
print(compare)
compare.to_csv("model_compare.csv", index=False)
best_name = compare.iloc[0]["model"]
print(f"\nEn iyi CV (TimeSeriesSplit) model: {best_name}")
```

CRISP-DM 5 – Değerlendirme (Evaluation)

Bu aşamada, oluşturulan **Lineer Regresyon modelinin** performansı hem eğitim (train) hem de test verisi üzerinde değerlendirilmiştir.
Amaç, modelin sıcaklık tahminlerinde ne kadar başarılı olduğunu belirlemek ve gelecekteki genelleme gücünü ölçmektir.

Değerlendirme Süreci

Model değerlendirmesi üç farklı ölçütle yapılmıştır:

- **R² (Determinasyon Katsayısı)** → Modelin verideki değişkenliği ne oranda açıkladığını gösterir.
- **MAE (Mean Absolute Error)** → Ortalama mutlak hata; tahmin ile gerçek değer farkının ortalaması.
- **RMSE (Root Mean Squared Error)** → Kareler ortalamasının karekökü; büyük hatalara daha fazla ceza verir.

Değerlendirme için hem **5-fold çapraz doğrulama (Cross-Validation)** hem de **test seti** kullanılmıştır.

Test Sonuçları

Metrik	Değer	Yorum
R²	**0.0562**	Model test verisindeki sıcaklık değişkenliğinin yaklaşık %5'ini açıklayabilmiştir. Bu değer, modelin genelleme başarısının düşük olduğunu gösterir.
MAE	**5.27 °C**	Ortalama tahmin hatası yaklaşık ±5 °C civarındadır.
RMSE	**6.15 °C**	Hata kareleri ortalamasının karekökü de 6 °C civarındadır; bu, bazı tahminlerin yüksek sapma gösterdiğini belirtir.

Sonuçların Yorumlanması

- Modelin **train verisindeki doğruluğu yüksek**, fakat test verisinde R²'nin düşmesi **aşırı öğrenme (overfitting)** olasılığına işaret etmektedir.
- Train (2013–2016) ve test (2017) dönemleri arasında **mevsimsel farklılık** ve olası **veri dağılımı kayması (data drift)** nedeniyle modelin test performansı zayıflamıştır.
- Özellikle **humidity** değişkeninin mevsime bağlı değişimi doğrusal modelin tahmin gücünü sınırlamıştır.

Artık (Residual) Analizi

Elde edilen artıklar (gerçek − tahmin) incelendiğinde:

- Artıkların ortalaması 0 civarındadır, bu da modelin sistematik yanlılık göstermediğini gösterir.
- Ancak varyansın yüksek olması, doğrusal modelin verideki karmaşık yapıyı tam olarak yakalayamadığını göstermektedir.

Histogramda artıklarda yaklaşık normal bir dağılım gözlenmiş, fakat uç sapmalar (±10 °C üzeri farklar) modelin iyileştirilebileceğini göstermektedir.

Model Karşılaştırma Sonuçları

Model	CV_R²	CV_RMSE	Yorum
GradientBoosting	**0.808**	**2.85**	Çok güçlü, iyi genelleşiyor
RandomForest	0.794	2.95	Benzer başarıda, stabil
DecisionTree	0.751	3.25	Basit ama iyi
SVR / KNN	0.57 civarı	3.6–3.8	Orta düzey, doğrusal olmayan etkileri kısmen yakalıyor
Lineer / Ridge / Lasso / ElasticNet	$R^2 < 0$	68 civarı	Tam başarısız (veri yapısı doğrusal değil)

Sonuç:
En iyi genel model: **GradientBoostingRegressor**

Train Sonuçları:

Metrik	Değer	Yorum
R^2	0.920	Model train verisinin %92'sini açıklıyor
MAE	1.54 °C	Ortalama hata ±1.5 °C civarında
RMSE	2.08 °C	Gayet düşük hata; tahminler stabil

Bu da demek oluyor ki model ne aşırı öğrenmiş (overfit) ne de zayıf kalmış — oldukça optimum bir sonuç.

Sonuç :

- Kod başarılı, tüm aşamalar doğru çalıştı.
- En iyi model: GradientBoostingRegressor (CV RMSE ≈ 2.83, Train R^2 ≈ 0.92).
- Lineer modellerin başarısız olması verideki doğrusal olmayan ve mevsimsel etkiler yüzünden beklenen bir durumdu.
- RandomForest da güçlü bir alternatif; ancak GradientBoosting daha az varyanslı sonuç veriyor.