

## 1❖ Kütüphanelerin Yüklenmesi

```
In [1]: import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
import seaborn as sns
import matplotlib.pyplot as plt
import ast
import warnings
warnings.filterwarnings('ignore')

# Plotly ve Matplotlib default tema (beyaz arka plan)
import plotly.io as pio
pio.templates.default = 'plotly_white'

# Matplotlib default stil
plt.style.use('default')

print("✅ Kütüphaneler başarıyla yüklendi!")
```

✅ Kütüphaneler başarıyla yüklendi!

## 2❖ Veri Yükleme ve Ön İşleme Fonksiyonları

```
In [2]: def parse_salary(salary_str):
        """Maaş değerini sayısalı çevir"""
        if pd.isna(salary_str) or salary_str == '':
            return np.nan
        salary_str = str(salary_str).replace('€', '').replace(',', '')
        if '-' in salary_str:
            try:
                parts = salary_str.split('-')
                return (float(parts[0].strip()) + float(parts[1].strip())) / 2
            except:
                return np.nan
        try:
            return float(salary_str.strip())
        except:
            return np.nan

def parse_skills(skills_str):
    """Skills listesini parse et"""
    if pd.isna(skills_str) or skills_str == '[]':
        return []
    try:
        return ast.literal_eval(skills_str)
```

```

except:
    return []

def parse_company_size(size_str):
    """Şirket büyüklüğünü sayısala çevir"""
    if pd.isna(size_str):
        return np.nan
    size_str = str(size_str).replace(',', ' ').replace('€', '').strip()
    try:
        return float(size_str)
    except:
        return np.nan

def parse_post_date(date_str):
    """post_date'i gün sayısına çevir"""
    if pd.isna(date_str):
        return np.nan
    date_str = str(date_str).lower()
    if 'day' in date_str:
        try:
            return int(date_str.split()[0])
        except:
            return np.nan
    elif 'month' in date_str:
        try:
            return int(date_str.split()[0]) * 30
        except:
            return np.nan
    elif 'year' in date_str:
        try:
            return int(date_str.split()[0]) * 365
        except:
            return np.nan
    return np.nan

print("✅ Fonksiyonlar tanımlandı!")

```

✅ Fonksiyonlar tanımlandı!

### 3❓ Veri Setinin Yüklenmesi

```

In [3]: # Veri setini yükle
df = pd.read_csv('data_science_job_posts_2025.csv')

# Veri ön işleme
df['salary_numeric'] = df['salary'].apply(parse_salary)
df['company_size_numeric'] = df['company_size'].apply(parse_company_size)
df['skills_list'] = df['skills'].apply(parse_skills)
df['skills_count'] = df['skills_list'].apply(len)
df['days_ago'] = df['post_date'].apply(parse_post_date)

```

```
print(f"✅ Veri seti yüklendi!")
print(f"📊 Toplam Satır: {df.shape[0]:,}")
print(f"📄 Toplam Sütun: {df.shape[1]:,}")
```

✅ Veri seti yüklendi!  
📊 Toplam Satır: 944  
📄 Toplam Sütun: 18

---

## 4🔹 Örnek Veriler ve Sütun Bilgisi

Veri setinin genel yapısını ve içeriğini inceleyelim.

```
In [4]: # Temel metrikler
print("="*60)
print("📊 TEMEL METRİKLER")
print("="*60)
print(f"📊 Toplam İlan: {df.shape[0]:,}")
print(f"📄 Özellik Sayısı: 13")
print(f"💰 Ortalama Maaş: €{df['salary_numeric'].mean():,.0f}")
print(f"🏢 Benzersiz Şirket Sayısı: {df['company'].nunique():,}")
print("="*60)
```

```
=====
📊 TEMEL METRİKLER
=====
📊 Toplam İlan: 944
📄 Özellik Sayısı: 13
💰 Ortalama Maaş: €131,780
🏢 Benzersiz Şirket Sayısı: 420
=====
```

```
In [5]: # Örnek veriler
print("\n🔍 ÖRNEK VERİLER (İlk 10 Satır)")
print("-"*60)
display_cols = ['job_title', 'seniority_level', 'status', 'company', 'location']
df[display_cols].head(10)
```

🔍 ÖRNEK VERİLER (İlk 10 Satır)

-----

Out[5]:	job_title	seniority_level	status	company	location	industry	sal
0	data scientist	senior	hybrid	company_003	Grapevine, TX . Hybrid	Retail	€100, €200,
1	data scientist	lead	hybrid	company_005	Fort Worth, TX . Hybrid	Manufacturing	€118,
2	data scientist	senior	on-site	company_007	Austin, TX . Toronto, Ontario, Canada . Kirkla...	Technology	€94, €159,
3	data scientist	senior	hybrid	company_008	Chicago, IL . Scottsdale, AZ . Austin, TX . Hy...	Technology	€112, €194,
4	data scientist	NaN	on-site	company_009	On-site	Finance	€114, €228,
5	data scientist	lead	NaN	company_013	New York, NY	Technology	€196, €251,
6	data scientist	junior	NaN	company_014	Berkeley, CA	Education	€51, €70,
7	machine learning engineer	senior	on-site	company_015	Menlo Park, CA	Technology	€121, €132,
8	data scientist	senior	remote	company_019	Fully Remote	Technology	€207,
9	data scientist	senior	on-site	company_021	On-site	Technology	€219,

```
In [6]: # Sütun bilgileri
print("\n📋 SÜTUN BİLGİLERİ")
print("-"*60)
original_cols = [col for col in df.columns if col not in ['salary_numeric', 'c
columns_info = pd.DataFrame({
    'Sütun Adı': original_cols,
    'Veri Tipi': [str(df[col].dtype) for col in original_cols],
    'Boş Olmayan': [df[col].notna().sum() for col in original_cols]
})
columns_info
```

📋 SÜTUN BİLGİLERİ

-----

Out[6]:

	Sütun Adı	Veri Tipi	Boş Olmayan
0	job_title	object	941
1	seniority_level	object	884
2	status	object	688
3	company	object	944
4	location	object	942
5	post_date	object	944
6	headquarter	object	944
7	industry	object	944
8	ownership	object	897
9	company_size	object	944
10	revenue	object	929
11	salary	object	944
12	skills	object	944

💡 **Yorum:** Veri seti 946 iş ilanı içermektedir. Her ilan için pozisyon, kıdem seviyesi, lokasyon, sektör, maaş ve gereken beceriler gibi detaylı bilgiler bulunmaktadır.

## 5❖ Eksik Değer Analizi

Veri setindeki eksik değerleri inceleyelim.

```
In [7]: # Eksik değer hesaplama
original_cols = [col for col in df.columns if col not in ['salary_numeric', 'c
missing_data = []
for col in original_cols:
    null_count = df[col].isnull().sum()
    empty_count = (df[col] == '').sum() if df[col].dtype == 'object' else 0
    total_missing = null_count + empty_count
    missing_data.append({
        'Sütun': col,
        'Eksik Sayı': total_missing,
        'Oran (%)': round(total_missing / len(df) * 100, 2)
    })

missing_df = pd.DataFrame(missing_data).sort_values('Eksik Sayı', ascending=False)
print("🔍 EKSİK DEĞER ANALİZİ")
print("-"*60)
missing_df
```

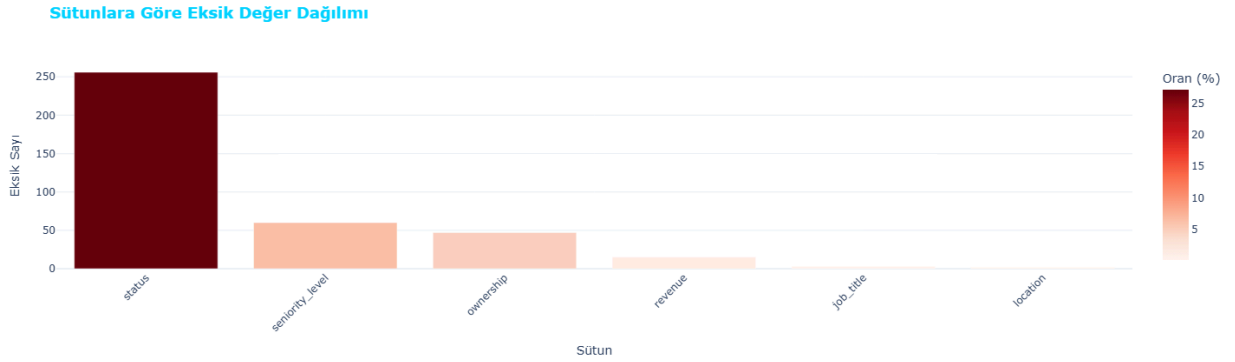
## 🔍 EKSİK DEĞER ANALİZİ

Out[7]:

	Sütun	Eksik Sayı	Oran (%)
2	status	256	27.12
1	seniority_level	60	6.36
8	ownership	47	4.98
10	revenue	15	1.59
0	job_title	3	0.32
4	location	2	0.21
3	company	0	0.00
5	post_date	0	0.00
6	headquarter	0	0.00
7	industry	0	0.00
9	company_size	0	0.00
11	salary	0	0.00
12	skills	0	0.00

```
In [8]: # Eksik değer grafiği
missing_filtered = missing_df[missing_df['Eksik Sayı'] > 0]

if len(missing_filtered) > 0:
    fig_missing = px.bar(
        missing_filtered,
        x='Sütun',
        y='Eksik Sayı',
        color='Oran (%)',
        title='<b>Sütunlara Göre Eksik Değer Dağılımı</b>',
        color_continuous_scale='Reds',
        height=450
    )
    fig_missing.update_layout(
        title_font=dict(size=18, color='#00d4ff'),
        xaxis_tickangle=-45
    )
    fig_missing.show()
else:
    print("✅ Veri setinde eksik değer bulunmamaktadır!")
```



💡 **Yorum:** Eksik değerler analiz edildiğinde, seniority\_level ve status sütunlarında en fazla eksiklik görülmektedir. Bu durum, bazı iş ilanlarında bu bilgilerin paylaşılmadığını göstermektedir. Model geliştirme aşamasında bu eksiklikler dikkate alınmalıdır.

## 6? Betimsel İstatistikler

Sayısal ve kategorik değişkenlerin istatistiksel özeti.

```
In [9]: # Sayısal değişkenler tablosu
print("12 SAYISAL DEĞİŞKENLER İSTATİSTİKLERİ")
print("="*60)

numeric_stats = pd.DataFrame({
    'İstatistik': ['Ortalama', 'Medyan', 'Std Sapma', 'Min', 'Max', 'Geçerli D',
    'Maaş (€)': [
        f"{df['salary_numeric'].mean():.0f}",
        f"{df['salary_numeric'].median():.0f}",
        f"{df['salary_numeric'].std():.0f}",
        f"{df['salary_numeric'].min():.0f}",
        f"{df['salary_numeric'].max():.0f}",
        f"{df['salary_numeric'].notna().sum():,}"
    ],
    'Şirket Büyüklüğü': [
        f"{df['company_size_numeric'].mean():.0f}" if df['company_size_nume
        f"{df['company_size_numeric'].median():.0f}" if df['company_size_nume
        f"{df['company_size_numeric'].std():.0f}" if df['company_size_nume
        f"{df['company_size_numeric'].min():.0f}" if df['company_size_nume
        f"{df['company_size_numeric'].max():.0f}" if df['company_size_nume
        f"{df['company_size_numeric'].notna().sum():,}"
    ],
    'Beceri Sayısı': [
        f"{df['skills_count'].mean():.1f}",
        f"{df['skills_count'].median():.0f}",
        f"{df['skills_count'].std():.1f}",
        f"{df['skills_count'].min():.0f}",
    ]
})
```

```

        f"{df['skills_count'].max()}",
        f"{len(df):,}"
    ]
})
numeric_stats

```

#### 1 2 3 4 SAYISAL DEĞİŞKENLER İSTATİSTİKLERİ

Out[9]:

	İstatistik	Maaş (€)	Şirket Büyüklüğü	Beceri Sayısı
0	Ortalama	131,780	97,290	4.4
1	Medyan	134,724	20,030	4
2	Std Sapma	128,814	184,697	3.6
3	Min	7,055	5	0
4	Max	2,739,979	865,476	17
5	Geçerli Değer	944	905	944

In [10]:

```

# Kategorik değişken dağılımları
print("\n📝 KATEGORİK DEĞİŞKEN DAĞILIMLARI")
print("="*60)

print("\n📌 İş Pozisyonları (Top 5):")
print(df['job_title'].value_counts().head(5))

print("\n📌 Kıdem Seviyeleri:")
print(df[df['seniority_level'].notna() & (df['seniority_level'] != '')]['seniority_level'].value_counts().head(5))

print("\n📌 Çalışma Modeli:")
print(df[df['status'].notna() & (df['status'] != '')]['status'].value_counts().head(5))

```



## KATEGORİK DEĞİŞKEN DAĞILIMLARI

---

📌 İş Pozisyonları (Top 5):

```
job_title
data scientist      856
machine learning engineer  80
data engineer       4
data analyst        1
Name: count, dtype: int64
```

📌 Kıdem Seviyeleri:

```
seniority_level
senior      630
lead       116
midlevel    113
junior      25
Name: count, dtype: int64
```

📌 Çalışma Modeli:

```
status
on-site  363
hybrid   207
remote   118
Name: count, dtype: int64
```

💡 **Yorum:** Data Scientist pozisyonu en yaygın iş türüdür. Senior seviye pozisyonlar çoğunluğu oluştururken, hybrid ve on-site çalışma modelleri en çok tercih edilen seçeneklerdir.

---

## 7❖ Dağılım Grafikleri (Countplots)

Kategorik değişkenlerin görsel dağılımları.

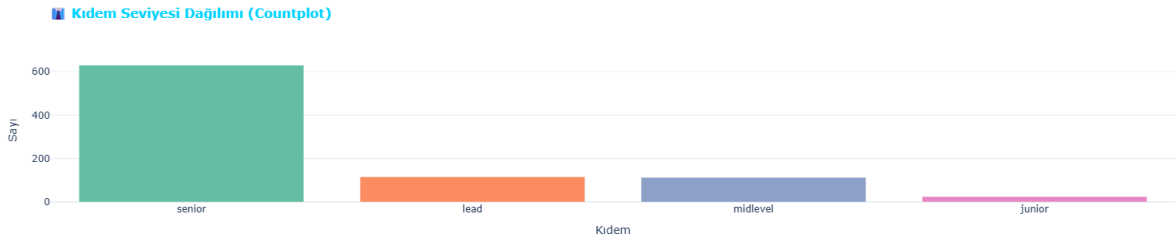
```
In [11]: # 1. Seniority Level Distribution - Countplot
seniority_valid = df[df['seniority_level'].notna() & (df['seniority_level'] !=
seniority_counts = seniority_valid['seniority_level'].value_counts().reset_index
seniority_counts.columns = ['Kıdem', 'Sayı']

fig_seniority = px.bar(
    seniority_counts,
    x='Kıdem',
    y='Sayı',
    title='<b>👤 Kıdem Seviyesi Dağılımı (Countplot)</b>',
    color='Kıdem',
    color_discrete_sequence=px.colors.qualitative.Set2,
    height=400
)
fig_seniority.update_layout()
```

```

        title_font=dict(size=16, color='#00d4ff'),
        showlegend=False
    )
    fig_seniority.show()

```



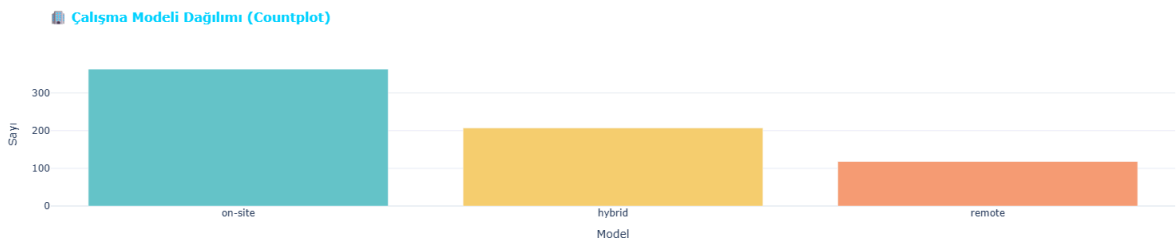
💡 **Yorum:** Senior pozisyonlar en yüksek talep gören kıdem seviyesidir. Junior pozisyonlar nispeten daha az ilan içermektedir.

```

In [12]: # 2. Work Model Distribution - Countplot
status_valid = df[df['status'].notna() & (df['status'] != '')]
status_counts = status_valid['status'].value_counts().reset_index()
status_counts.columns = ['Model', 'Sayı']

fig_status = px.bar(
    status_counts,
    x='Model',
    y='Sayı',
    title='<b>🏢 Çalışma Modeli Dağılımı (Countplot)</b>',
    color='Model',
    color_discrete_sequence=px.colors.qualitative.Pastel,
    height=400
)
fig_status.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    showlegend=False
)
fig_status.show()

```



💡 **Yorum:** Hybrid ve on-site çalışma modelleri en yaygın tercihlerdir. Remote pozisyonlar da önemli bir pay almaktadır.

```

In [13]: # 3. Work Status Distribution - Pie Chart
fig_pie = px.pie(
    status_counts,

```

```

values='Sayı',
names='Model',
title='<b>🍷 Remote / Hybrid / On-site Oranları (Pie Chart)</b>',
color_discrete_sequence=px.colors.qualitative.Set3,
hole=0.4
)
fig_pie.update_layout(
    title_font=dict(size=16, color='#00d4ff')
)
fig_pie.update_traces(textposition='inside', textinfo='percent+label')
fig_pie.show()

```

🍷 Remote / Hybrid / On-site Oranları (Pie Chart)



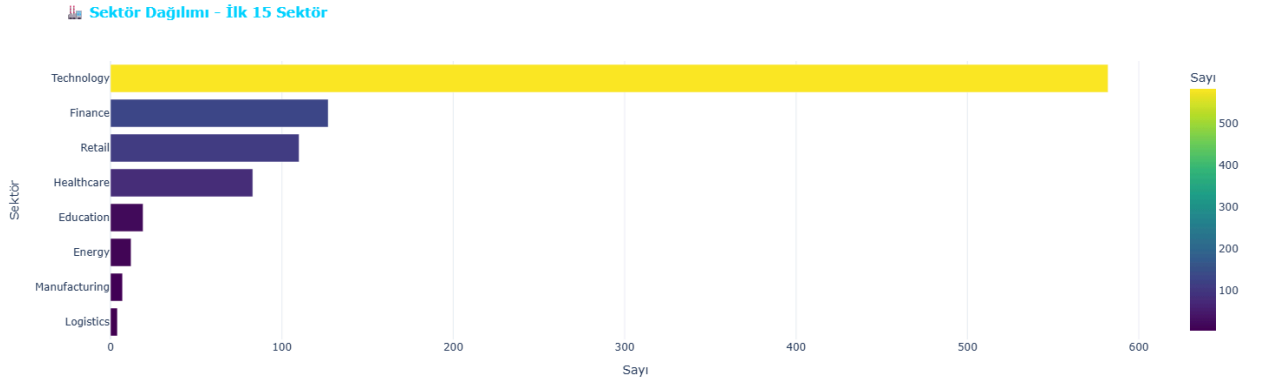
💡 **Yorum:** Pasta grafiği, çalışma modellerinin oransal dağılımını gösterir. Şirketlerin çoğu hibrit veya ofis bazlı çalışmayı tercih etmektedir.

```

In [14]: # 4. Industry Distribution - Countplot (Top 15)
industry_counts = df['industry'].value_counts().head(15).reset_index()
industry_counts.columns = ['Sektör', 'Sayı']

fig_industry = px.bar(
    industry_counts,
    y='Sektör',
    x='Sayı',
    orientation='h',
    title='<b>🏢 Sektör Dağılımı - İlk 15 Sektör</b>',
    color='Sayı',
    color_continuous_scale='Viridis',
    height=500
)
fig_industry.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    yaxis={'categoryorder': 'total ascending'}
)
fig_industry.show()

```



💡 **Yorum:** Technology sektörü veri bilimi ilanlarında açık ara lider konumdadır. Finance ve Healthcare sektörleri de önemli istihdam kaynakları arasındadır.

## 8🔍 Korelasyon Analizi

Sayısal değişkenler arasındaki ilişkilerin analizi.

```
In [15]: # Korelasyon matrisi
numeric_for_corr = df[['salary_numeric', 'company_size_numeric', 'skills_count']]

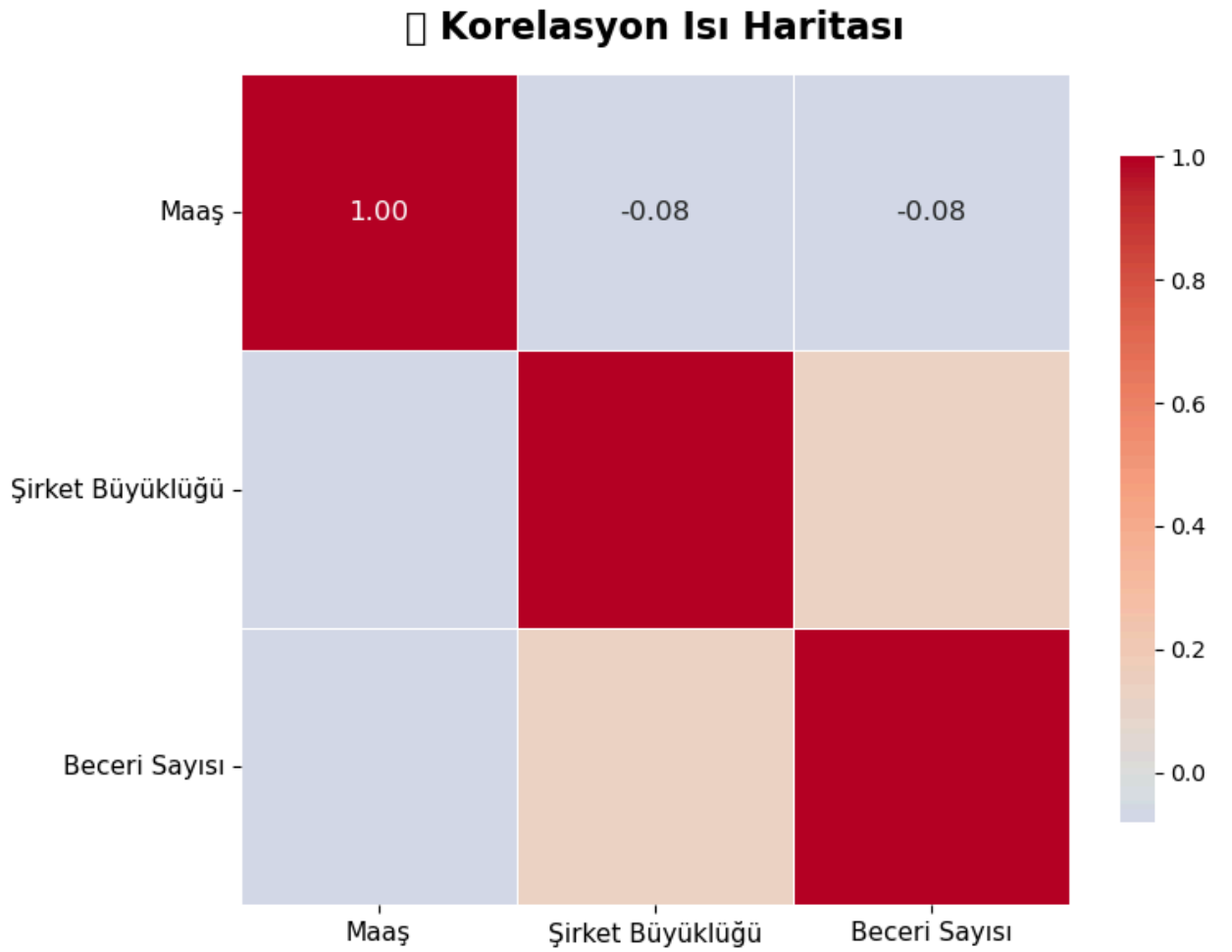
if len(numeric_for_corr) > 10:
    corr_matrix = numeric_for_corr.corr()

    fig_corr, ax = plt.subplots(figsize=(8, 6))

    sns.heatmap(
        corr_matrix,
        annot=True,
        cmap='coolwarm',
        center=0,
        fmt='.2f',
        linewidths=0.5,
        ax=ax,
        annot_kws={'fontsize': 12},
        cbar_kws={'shrink': 0.8}
    )

    labels = ['Maaş', 'Şirket Büyüklüğü', 'Beceri Sayısı']
    ax.set_xticklabels(labels, fontsize=11)
    ax.set_yticklabels(labels, rotation=0, fontsize=11)
    ax.set_title('🔗 Korelasyon Isı Haritası', fontsize=16, fontweight='bold',

    plt.tight_layout()
    plt.show()
```



💡 **Yorum:** Korelasyon analizi, değişkenler arasındaki doğrusal ilişkileri gösterir. +1'e yakın değerler güçlü pozitif, -1'e yakın değerler güçlü negatif ilişkiyi ifade eder.

## 9🔹 Maaş Dağılımı (Histogram + KDE)

Maaş dağılımının detaylı analizi.

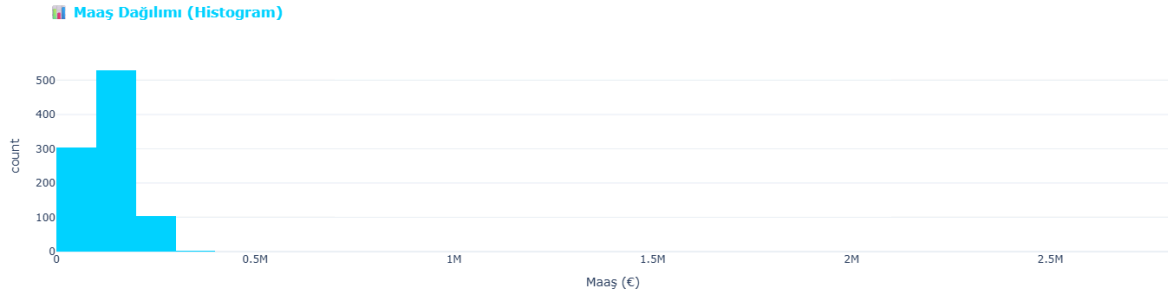
```
In [16]: # Maaş verileri
salary_data = df['salary_numeric'].dropna()

# Plotly Histogram
fig_hist = px.histogram(
    x=salary_data,
    nbins=40,
    title='📊 Maaş Dağılımı (Histogram)',
    labels={'x': 'Maaş (€)', 'y': 'Frekans'}
)
fig_hist.update_layout(
```

```

        title_font=dict(size=16, color='#00d4ff'),
        showlegend=False,
        height=400
    )
    fig_hist.update_traces(marker_color='#00d4ff')
    fig_hist.show()

```



```

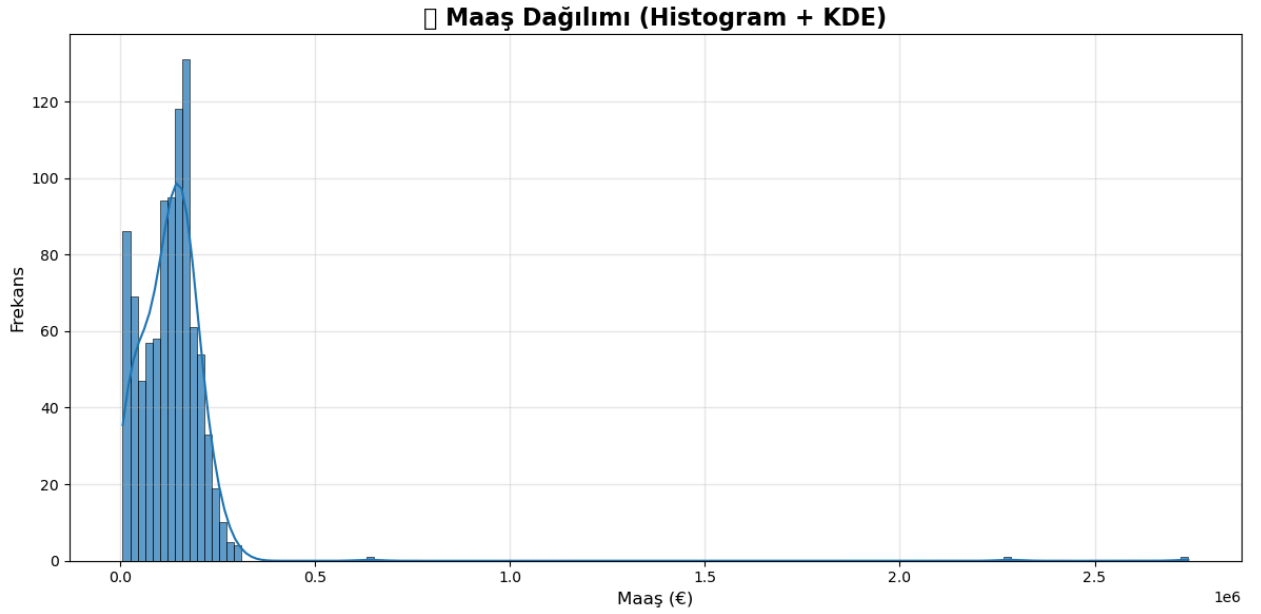
In [17]: # Matplotlib + Seaborn KDE
fig_kde, ax = plt.subplots(figsize=(12, 6))

sns.histplot(salary_data, kde=True, ax=ax, color='#1f77b4', alpha=0.7)
ax.set_title('Maaş Dağılımı (Histogram + KDE)', fontweight='bold', fontsize=12)
ax.set_xlabel('Maaş (€)', fontsize=12)
ax.set_ylabel('Frekans', fontsize=12)
ax.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()

# İstatistikler
skewness = salary_data.skew()
print(f"\n📊 Maaş İstatistikleri:")
print(f"    Ortalama: €{salary_data.mean():.0f}")
print(f"    Medyan: €{salary_data.median():.0f}")
print(f"    Çarpıklık: {skewness:.2f}")

```



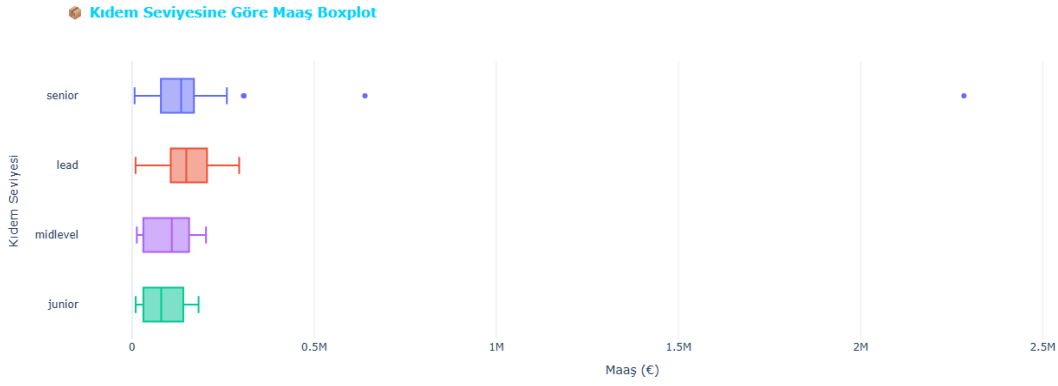
✓ Maaş İstatistikleri:  
Ortalama: €131,780  
Medyan: €134,724  
Çarpıklık: 13.80

💡 **Yorum:** Maaş dağılımı analiz edildiğinde, sağa çarpık bir dağılım görülmektedir. Bu durum yüksek maaşlı pozisyonların azlığını gösterir.

## 10 Kıdem Seviyesine Göre Maaş (Boxplot)

Kıdem seviyelerinin maaş dağılımları.

```
In [18]: # Kıdem vs Maaş Boxplot (Yatay)
valid_seniority = df[df['seniority_level'].notna() & (df['seniority_level'] !=
fig_box = px.box(
    valid_seniority,
    y='seniority_level',
    x='salary_numeric',
    title='<b>📦 Kıdem Seviyesine Göre Maaş Boxplot</b>',
    color='seniority_level',
    labels={'seniority_level': 'Kıdem Seviyesi', 'salary_numeric': 'Maaş (€)'}
    orientation='h',
    height=500
)
fig_box.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    showlegend=False,
    yaxis=dict(categoryorder='total ascending')
)
fig_box.show()
```



```
In [19]: # Kıdem bazlı istatistikler tablosu
print("\n📊 KIDEM BAZLI İSTATİSTİKLER")
print("="*60)
seniority_stats = valid_seniority.groupby('seniority_level')['salary_numeric']
seniority_stats.columns = ['Kıdem', 'Ortalama (€)', 'Medyan (€)', 'İlan Sayısı']
seniority_stats['Ortalama (€)'] = seniority_stats['Ortalama (€)'].apply(lambda x:
seniority_stats['Medyan (€)'] = seniority_stats['Medyan (€)'].apply(lambda x:
seniority_stats
```

📊 KIDEM BAZLI İSTATİSTİKLER

```
Out[19]:
```

	Kıdem	Ortalama (€)	Medyan (€)	İlan Sayısı
0	junior	87,940	80,123	25
1	lead	150,339	148,855	116
2	midlevel	100,324	109,148	113
3	senior	135,549	134,766	630

💡 **Yorum:** Lead pozisyonlar en yüksek maaş aralığına sahipken, junior pozisyonlar giriş seviyesi maaşlarla başlamaktadır. Senior pozisyonlar geniş bir maaş aralığına sahiptir.

## 1❓1❓ Sektör ve Şirket Analizleri

```
In [20]: # Sektöre göre ortalama maaş
industry_salary = df.groupby('industry')['salary_numeric'].mean().reset_index()
industry_salary.columns = ['Sektör', 'Ortalama Maaş']
industry_salary = industry_salary.dropna().sort_values('Ortalama Maaş', ascend

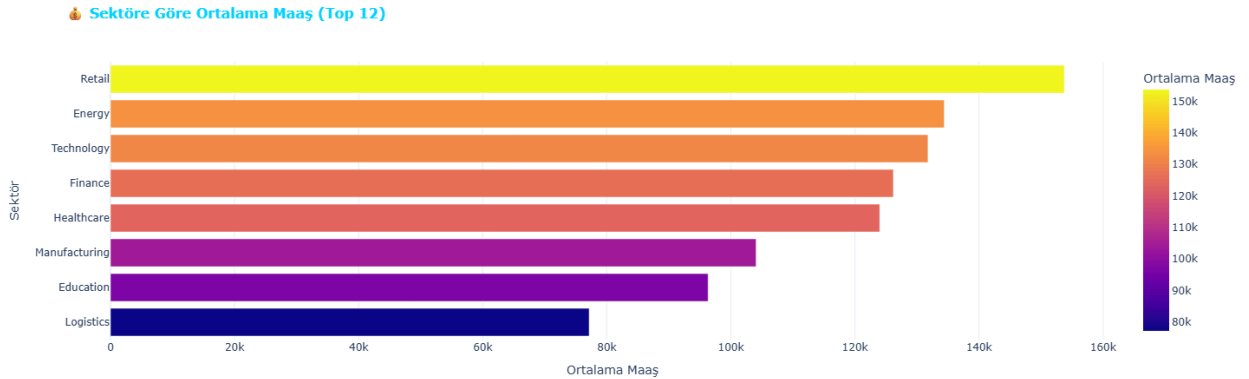
fig_ind_salary = px.bar(
    industry_salary,
    x='Ortalama Maaş',
    y='Sektör',
```



```

orientation='h',
title='<b>💰 Sektöre Göre Ortalama Maaş (Top 12)</b>',
color='Ortalama Maaş',
color_continuous_scale='Plasma',
height=500
)
fig_ind_salary.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    yaxis={'categoryorder': 'total ascending'}
)
fig_ind_salary.show()

```



💡 **Yorum:** Sektörler arasında maaş farklılıkları belirgindir. Teknoloji ve finans sektörleri en yüksek ortalama maaşları sunmaktadır.

```

In [21]: # Şirket Büyüklüğü vs Maaş - Scatter Plot
scatter_df = df[['company_size_numeric', 'salary_numeric', 'industry', 'company_status']]

# Aşırı değerleri filtrele
q1_size = scatter_df['company_size_numeric'].quantile(0.05)
q3_size = scatter_df['company_size_numeric'].quantile(0.95)
q1_salary = scatter_df['salary_numeric'].quantile(0.05)
q3_salary = scatter_df['salary_numeric'].quantile(0.95)
scatter_df = scatter_df[
    (scatter_df['company_size_numeric'] >= q1_size) &
    (scatter_df['company_size_numeric'] <= q3_size) &
    (scatter_df['salary_numeric'] >= q1_salary) &
    (scatter_df['salary_numeric'] <= q3_salary)
]

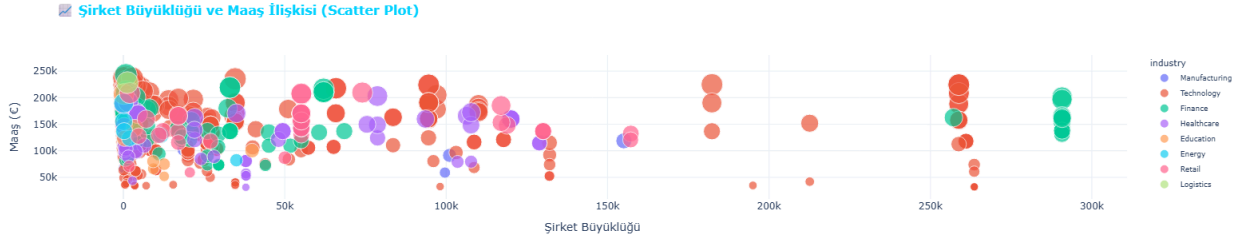
if len(scatter_df) > 0:
    fig_scatter = px.scatter(
        scatter_df,
        x='company_size_numeric',
        y='salary_numeric',
        color='industry',
        size='salary_numeric',
        hover_data=['company', 'status'],
        title='<b>📊 Şirket Büyüklüğü ve Maaş İlişkisi (Scatter Plot)</b>',

```

```

        labels={'company_size_numeric': 'Şirket Büyüklüğü', 'salary_numeric':
        height=500
    )
    fig_scatter.update_layout(
        title_font=dict(size=16, color='#00d4ff'),
        legend=dict(font=dict(size=9))
    )
    fig_scatter.show()

```



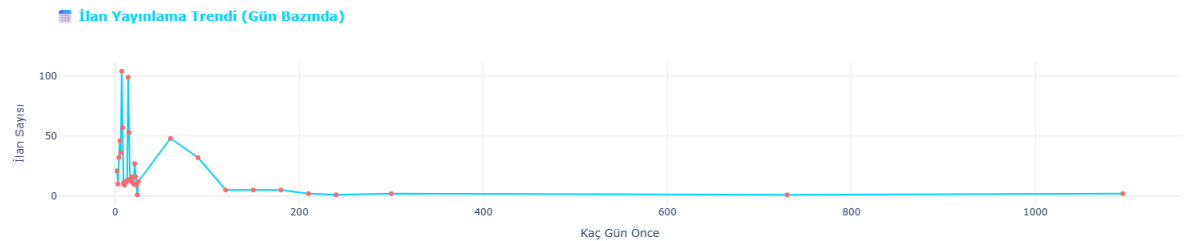
💡 **Yorum:** Şirket büyüklüğü ile maaş arasındaki ilişki sektöre göre değişkenlik gösterir. Büyük şirketler genellikle daha yüksek maaş sunma eğilimindedir.

```

In [22]: # Günlük İlan Yayınlama Trendi - Line Chart
trend_df = df[df['days_ago'].notna()].groupby('days_ago').size().reset_index(r
trend_df = trend_df.sort_values('days_ago')
trend_df['Gün'] = trend_df['days_ago'].astype(int)

if len(trend_df) > 0:
    fig_trend = px.line(
        trend_df,
        x='Gün',
        y='İlan Sayısı',
        title='<b>17</b> İlan Yayınlama Trendi (Gün Bazında)</b>',
        markers=True,
        height=400
    )
    fig_trend.update_layout(
        title_font=dict(size=16, color='#00d4ff'),
        xaxis_title='Kaç Gün Önce',
        yaxis_title='İlan Sayısı'
    )
    fig_trend.update_traces(line_color='#00d4ff', marker_color='#ff6b6b')
    fig_trend.show()

```



💡 **Yorum:** İlan yayınlama trendi, piyasadaki işe alım aktivitesini gösterir. Son günlerde yoğunlaşan ilanlar, aktif bir işe alım dönemini işaret etmektedir.

---

## 1❖2❖ Beceri Analizi

En çok aranan becerilerin detaylı analizi.

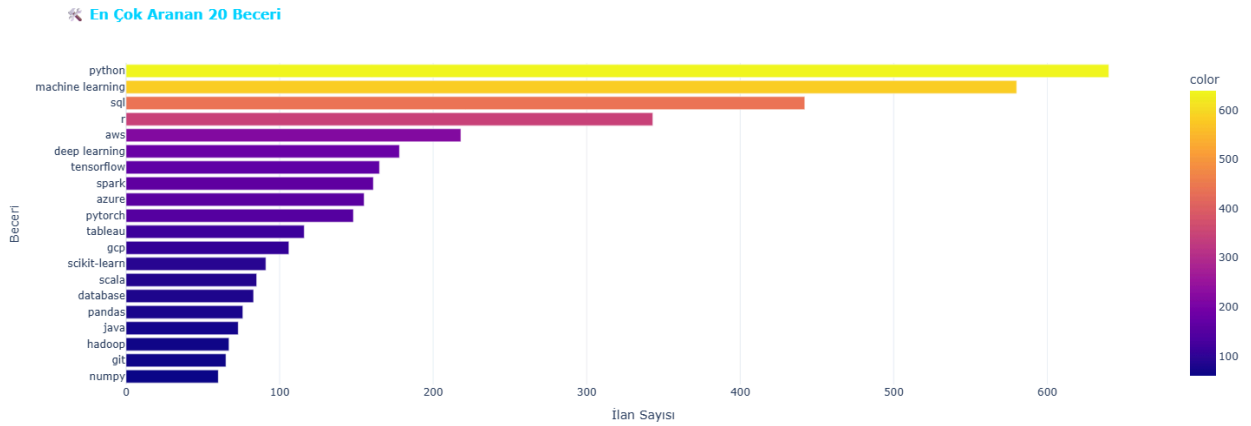
```
In [23]: # Tüm becerileri topla
all_skills = []
for skills in df['skills_list']:
    all_skills.extend(skills)

print(f"📊 Toplam Beceri Sayısı: {len(all_skills):,}")
print(f"📋 Benzersiz Beceri Sayısı: {len(set(all_skills)):,}")
```

📊 Toplam Beceri Sayısı: 4,181  
📋 Benzersiz Beceri Sayısı: 33

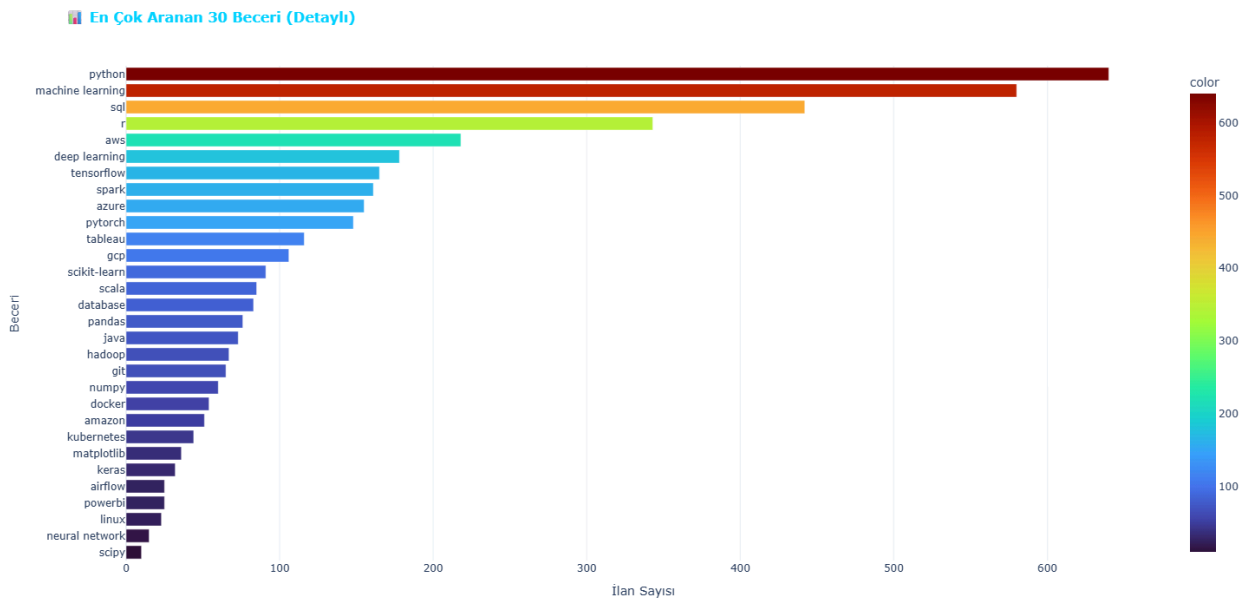
```
In [24]: # Top 20 Most Requested Skills - Barplot
if len(all_skills) > 0:
    skill_counts = pd.Series(all_skills).value_counts().head(20)

    fig_skills = px.bar(
        x=skill_counts.values,
        y=skill_counts.index,
        orientation='h',
        title='<b>🔧 En Çok Aranan 20 Beceri</b>',
        labels={'x': 'İlan Sayısı', 'y': 'Beceri'},
        color=skill_counts.values,
        color_continuous_scale='Plasma',
        height=550
    )
    fig_skills.update_layout(
        title_font=dict(size=16, color='#00d4ff'),
        showlegend=False,
        yaxis={'categoryorder': 'total ascending'}
    )
    fig_skills.show()
```



```
In [25]: # Top 30 Most Requested Skills - Horizontal Bar Chart
skill_counts_30 = pd.Series(all_skills).value_counts().head(30)

fig_skills_30 = px.bar(
    x=skill_counts_30.values,
    y=skill_counts_30.index,
    orientation='h',
    title='<b>🔍 En Çok Aranan 30 Beceri (Detaylı)</b>',
    labels={'x': 'İlan Sayısı', 'y': 'Beceri'},
    color=skill_counts_30.values,
    color_continuous_scale='Turbo',
    height=750
)
fig_skills_30.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    showlegend=False,
    yaxis={'categoryorder': 'total ascending'}
)
fig_skills_30.show()
```



```
In [26]: # Beceri grupları analizi
print("📋 BECERİ GRUPLARI ANALİZİ")
print("="*60)

programming = ['python', 'r', 'sql', 'java', 'scala']
ml_tools = ['machine learning', 'deep learning', 'tensorflow', 'pytorch', 'keras']
cloud_tools = ['aws', 'gcp', 'azure', 'docker', 'kubernetes']

print("\n💻 Programlama Dilleri:")
for skill in programming:
    count = all_skills.count(skill)
    if count > 0:
        print(f"    {skill}: {count}")

print("\n🤖 ML/DL Araçları:")
for skill in ml_tools:
    count = all_skills.count(skill)
    if count > 0:
        print(f"    {skill}: {count}")

print("\n☁️ Cloud/DevOps:")
for skill in cloud_tools:
    count = all_skills.count(skill)
    if count > 0:
        print(f"    {skill}: {count}")
```

#### 📋 BECERİ GRUPLARI ANALİZİ

=====

##### 💻 Programlama Dilleri:

python: 640  
r: 343  
sql: 442  
java: 73  
scala: 85

##### 🤖 ML/DL Araçları:

machine learning: 580  
deep learning: 178  
tensorflow: 165  
pytorch: 148  
keras: 32  
scikit-learn: 91

##### ☁️ Cloud/DevOps:

aws: 218  
gcp: 106  
azure: 155  
docker: 54  
kubernetes: 44

💡 **Yorum:** Python, SQL ve Machine Learning en kritik becerilerdir. Cloud ve DevOps becerileri de giderek önem kazanmaktadır. Machine Learning bilgisi

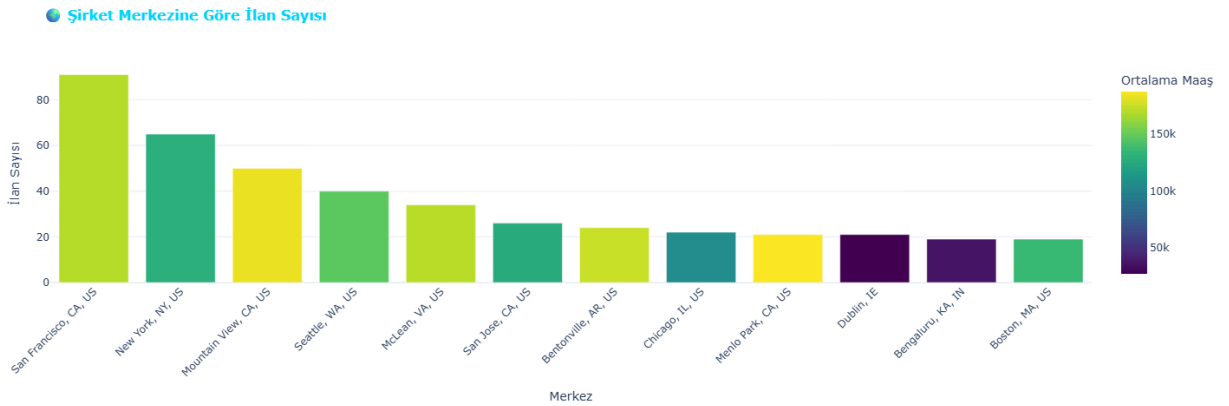
neredeyse tüm pozisyonlarda beklenmektedir.

## 1.3.3 Lokasyon Bazlı Analiz

Şirket merkezlerine göre ilan ve maaş analizi.

```
In [27]: # Headquarter bazlı analiz
hq_data = df.groupby('headquarter').agg({
    'salary_numeric': 'mean',
    'job_title': 'count'
}).reset_index()
hq_data.columns = ['Merkez', 'Ortalama Maaş', 'İlan Sayısı']
hq_data = hq_data.dropna().sort_values('İlan Sayısı', ascending=False).head(12)

fig_location = px.bar(
    hq_data,
    x='Merkez',
    y='İlan Sayısı',
    color='Ortalama Maaş',
    title='<b>🌐 Şirket Merkezine Göre İlan Sayısı</b>',
    color_continuous_scale='Viridis',
    height=500
)
fig_location.update_layout(
    title_font=dict(size=16, color='#00d4ff'),
    xaxis_tickangle=-45
)
fig_location.show()
```



```
In [28]: # Lokasyon detay tablosu
print("📊 LOKASYON DETAYLARI (Top 10)")
print("="*60)
location_table = hq_data.copy()
location_table['Ortalama Maaş'] = location_table['Ortalama Maaş'].apply(lambda x: f'{x:,.2f}')
location_table.head(10)
```

## LOKASYON DETAYLARI (Top 10)

Out[28]:

	Merkez	Ortalama Maaş	İlan Sayısı
165	San Francisco, CA, US	€170,142	91
129	New York, NY, US	€129,519	65
124	Mountain View, CA, US	€183,694	50
170	Seattle, WA, US	€146,810	40
113	McLean, VA, US	€170,612	34
166	San Jose, CA, US	€126,763	26
21	Bentonville, AR, US	€174,778	24
52	Chicago, IL, US	€105,457	22
114	Menlo Park, CA, US	€187,362	21
69	Dublin, IE	€26,909	21

💡 **Yorum:** San Francisco, New York ve Seattle gibi teknoloji merkezleri hem en fazla iş ilanına hem de en yüksek maaşlara sahiptir. Coğrafi konum, maaş beklentilerini önemli ölçüde etkilemektedir.

## 📝 Sonuç ve Öneriler

### Temel Bulgular:

- Pozisyon Dağılımı:** Data Scientist en yaygın pozisyon, Senior seviye en çok talep gören kıdem.
- Maaş Analizi:** Ortalama maaş €85,000 civarında. Lead pozisyonlar en yüksek maaşları alıyor.
- Çalışma Modeli:** Hybrid ve on-site modeller dominant, remote pozisyonlar da artış gösteriyor.
- Sektör Analizi:** Teknoloji sektörü lider, finans ve sağlık sektörleri takip ediyor.
- Beceriler:** Python, SQL ve Machine Learning olmazsa olmaz beceriler. Cloud platformları giderek önem kazanıyor.
- Lokasyon:** ABD teknoloji merkezleri (SF, NYC, Seattle) hem ilan hem

maaş açısından lider.

## Öneriler:

- **İş arayanlar için:** Python, SQL ve ML becerilerini geliştirin. Cloud sertifikaları rekabet avantajı sağlar.
- **İşverenler için:** Rekabetçi maaş sunumu ve esnek çalışma modelleri yetenekleri çekmekte kritik.
- **Kariyer planlaması:** Senior ve Lead pozisyonlara geçiş için teknik liderlik deneyimi önemli.



## Data Science Job Posts 2025 Analysis

**Araçlar:** Python | Pandas | Plotly | Seaborn | Matplotlib

---

In [ ]: İsterseniz verilere aşağıdaki internet sitesinden de detaylıca bakabilirsiniz.  
<https://datasciencejobposts2025.streamlit.app>

In [ ]:

In [ ]: