AIPROJECT

September 5, 2025

[1]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as pyt
import tensorflow as tf
```

[2]:
```
pwd
```

[2]:
```
'C:\\Users\\lenovo\\Desktop'
```

[3]:
```
import pandas as pd
df = pd.read_csv(r"C:\Users\lenovo\Downloads\ai_job_dataset.csv")
print(df.head())
```

```
    job_id              job_title  salary_usd salary_currency  \
0  AI00001  AI Research Scientist       90376             USD
1  AI00002   AI Software Engineer       61895             USD
2  AI00003          AI Specialist      152626             USD
3  AI00004           NLP Engineer       80215             USD
4  AI00005          AI Consultant       54624             EUR

  experience_level employment_type company_location company_size  \
0               SE              CT            China            M
1               EN              CT           Canada            M
2               MI              FL      Switzerland            L
3               SE              FL            India            M
4               EN              PT           France            S

  employee_residence  remote_ratio  \
0              China            50
1            Ireland           100
2        South Korea             0
3              India            50
4          Singapore           100

                                required_skills education_required  \
0         Tableau, PyTorch, Kubernetes, Linux, NLP           Bachelor
1  Deep Learning, AWS, Mathematics, Python, Docker             Master
```

```
2     Kubernetes, Deep Learning, Java, Hadoop, NLP          Associate
3                         Scala, SQL, Linux, Python               PhD
4                       MLOps, Java, Tableau, Python             Master

   years_experience    industry posting_date application_deadline  \
0                 9  Automotive   2024-10-18           2024-11-07
1                 1       Media   2024-11-20           2025-01-11
2                 2   Education   2025-03-18           2025-04-07
3                 7  Consulting   2024-12-23           2025-02-24
4                 0       Media   2025-04-15           2025-06-23

   job_description_length  benefits_score       company_name
0                    1076             5.9    Smart Analytics
1                    1268             5.2       TechCorp Inc
2                    1974             9.4    Autonomous Tech
3                    1345             8.6     Future Systems
4                    1989             6.6  Advanced Robotics
```

[4]:
```
df.head()
```

[4]:
```
     job_id              job_title  salary_usd salary_currency  \
0  AI00001  AI Research Scientist       90376             USD
1  AI00002   AI Software Engineer       61895             USD
2  AI00003          AI Specialist      152626             USD
3  AI00004           NLP Engineer       80215             USD
4  AI00005          AI Consultant       54624             EUR

  experience_level employment_type company_location company_size  \
0               SE              CT            China            M
1               EN              CT           Canada            M
2               MI              FL      Switzerland            L
3               SE              FL            India            M
4               EN              PT           France            S

  employee_residence  remote_ratio  \
0              China            50
1            Ireland           100
2        South Korea             0
3              India            50
4          Singapore           100

                                  required_skills education_required  \
0        Tableau, PyTorch, Kubernetes, Linux, NLP           Bachelor
1  Deep Learning, AWS, Mathematics, Python, Docker             Master
2    Kubernetes, Deep Learning, Java, Hadoop, NLP          Associate
3                        Scala, SQL, Linux, Python               PhD
4                      MLOps, Java, Tableau, Python             Master
```

```
   years_experience    industry posting_date application_deadline  \
0                 9  Automotive   2024-10-18           2024-11-07
1                 1       Media   2024-11-20           2025-01-11
2                 2   Education   2025-03-18           2025-04-07
3                 7  Consulting   2024-12-23           2025-02-24
4                 0       Media   2025-04-15           2025-06-23

   job_description_length  benefits_score       company_name
0                    1076             5.9    Smart Analytics
1                    1268             5.2       TechCorp Inc
2                    1974             9.4    Autonomous Tech
3                    1345             8.6     Future Systems
4                    1989             6.6  Advanced Robotics
```

[5]:
```
df.tail()
```

[5]:
```
        job_id                    job_title  salary_usd salary_currency  \
14995  AI14996            Robotics Engineer       38604             USD
14996  AI14997  Machine Learning Researcher       57811             GBP
14997  AI14998                 NLP Engineer      189490             USD
14998  AI14999                   Head of AI       79461             EUR
14999  AI15000     Computer Vision Engineer       56481             USD

      experience_level employment_type company_location company_size  \
14995               EN              FL          Finland            S
14996               EN              CT   United Kingdom            M
14997               EX              CT      South Korea            L
14998               EN              FT      Netherlands            M
14999               MI              PT          Austria            S

      employee_residence  remote_ratio  \
14995            Finland            50
14996     United Kingdom             0
14997        South Korea            50
14998        Netherlands             0
14999            Austria            50

                                  required_skills education_required  \
14995                       Java, Kubernetes, Azure           Bachelor
14996       Mathematics, Docker, SQL, Deep Learning             Master
14997                            Scala, Spark, NLP          Associate
14998     Java, Computer Vision, Python, TensorFlow                PhD
14999  Scala, Azure, Deep Learning, GCP, Mathematics                PhD

       years_experience       industry posting_date application_deadline  \
14995                 1         Energy   2025-02-06           2025-03-25
```

```
14996                 0     Government   2024-10-16           2024-10-30
14997                17  Manufacturing   2024-03-19           2024-05-02
14998                 1    Real Estate   2024-03-22           2024-04-23
14999                 2     Technology   2024-07-18           2024-08-10

       job_description_length  benefits_score       company_name
14995                    1635             7.9  Advanced Robotics
14996                    1624             8.2    Smart Analytics
14997                    1336             7.4     AI Innovations
14998                    1935             5.6    Smart Analytics
14999                    2492             7.6     AI Innovations
```

[6]:
```
df.describe()
```

[6]:
```
          salary_usd  remote_ratio  years_experience  job_description_length  \
count   15000.000000  15000.000000      15000.000000            15000.000000
mean   115348.965133     49.483333          6.253200             1503.314733
std     60260.940438     40.812712          5.545768              576.127083
min     32519.000000      0.000000          0.000000              500.000000
25%     70179.750000      0.000000          2.000000             1003.750000
50%     99705.000000     50.000000          5.000000             1512.000000
75%    146408.500000    100.000000         10.000000             2000.000000
max    399095.000000    100.000000         19.000000             2499.000000

       benefits_score
count    15000.000000
mean         7.504273
std          1.450870
min          5.000000
25%          6.200000
50%          7.500000
75%          8.800000
max         10.000000
```

[7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 19 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   job_id                  15000 non-null  object
 1   job_title               15000 non-null  object
 2   salary_usd              15000 non-null  int64
 3   salary_currency         15000 non-null  object
 4   experience_level        15000 non-null  object
 5   employment_type         15000 non-null  object
 6   company_location        15000 non-null  object
```

```
 7   company_size            15000 non-null  object
 8   employee_residence      15000 non-null  object
 9   remote_ratio            15000 non-null  int64
 10  required_skills         15000 non-null  object
 11  education_required      15000 non-null  object
 12  years_experience        15000 non-null  int64
 13  industry                15000 non-null  object
 14  posting_date            15000 non-null  object
 15  application_deadline    15000 non-null  object
 16  job_description_length  15000 non-null  int64
 17  benefits_score          15000 non-null  float64
 18  company_name            15000 non-null  object
dtypes: float64(1), int64(4), object(14)
memory usage: 2.2+ MB
```

[8]:
```
df.isnull().any()
```

[8]:
```
job_id                  False
job_title               False
salary_usd              False
salary_currency         False
experience_level        False
employment_type         False
company_location        False
company_size            False
employee_residence      False
remote_ratio            False
required_skills         False
education_required      False
years_experience        False
industry                False
posting_date            False
application_deadline    False
job_description_length  False
benefits_score          False
company_name            False
dtype: bool
```

[9]:
```
df.isnull().sum()
```

[9]:
```
job_id                  0
job_title               0
salary_usd              0
salary_currency         0
experience_level        0
employment_type         0
company_location        0
company_size            0
```

```
employee_residence        0
remote_ratio              0
required_skills           0
education_required        0
years_experience          0
industry                  0
posting_date              0
application_deadline      0
job_description_length    0
benefits_score            0
company_name              0
dtype: int64
```

[10]:
```
df.dropna(axis=0, inplace=True)
```

[11]:
```
df['posting_date'] = pd.to_datetime(df['posting_date'], errors='coerce')
df['year'] = df['posting_date'].dt.year
```

[12]:
```
df[['posting_date', 'year']].head()
```

[12]:
```
  posting_date  year
0   2024-10-18  2024
1   2024-11-20  2024
2   2025-03-18  2025
3   2024-12-23  2024
4   2025-04-15  2025
```

[13]:
```
salary_time_series = (
    df.groupby(['year', 'job_title'])['salary_usd']
    .mean()
    .reset_index()
    .sort_values(by=['job_title', 'year'])
)
```

[14]:
```
salary_time_series.head()
```

[14]:
```
    year           job_title     salary_usd
0   2024        AI Architect  117231.607383
20  2025        AI Architect  118134.365714
1   2024       AI Consultant  113095.028902
21  2025       AI Consultant  115290.145946
2   2024  AI Product Manager  112609.448097
```

[15]:
```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression
```

[16]:
```
X = df.drop(columns=["salary_usd"])
y = df["salary_usd"]
print(df.columns)
```

```
Index(['job_id', 'job_title', 'salary_usd', 'salary_currency',
       'experience_level', 'employment_type', 'company_location',
       'company_size', 'employee_residence', 'remote_ratio', 'required_skills',
       'education_required', 'years_experience', 'industry', 'posting_date',
       'application_deadline', 'job_description_length', 'benefits_score',
       'company_name', 'year'],
      dtype='object')
```

[17]:
```
# Step 1: Define target and features
X = df.drop(columns=["salary_usd"])
y = df["salary_usd"]

# Step 2: Split categorical & numeric columns
categorical_cols = X.select_dtypes(include=["object"]).columns.tolist()
numeric_cols = X.select_dtypes(include=["int64", "float64"]).columns.tolist()

# Step 3: Preprocessor
preprocessor = ColumnTransformer([
    ("cat", OneHotEncoder(handle_unknown="ignore", max_categories=20),␣
 ↪categorical_cols),
    ("num", StandardScaler(), numeric_cols)
])

# Step 4: Pipeline with Linear Regression
model = Pipeline([
    ("preprocessor", preprocessor),
    ("regressor", LinearRegression())
])

# Step 5: Train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Step 6: Fit and evaluate
model.fit(X_train, y_train)

from sklearn.metrics import mean_squared_error, r2_score
y_pred = model.predict(X_test)

print("R² Score:", r2_score(y_test, y_pred))
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

```
R² Score: 0.8437573765826831
```

```
RMSE: 23871.453605796432

C:\ProgramData\anaconda3\Lib\site-packages\sklearn\metrics\_regression.py:492:
FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in
1.6. To calculate the root mean squared error, use the
function'root_mean_squared_error'.
  warnings.warn(
```

[18]:
```
!pip install xgboost
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: xgboost in
c:\users\lenovo\appdata\roaming\python\python312\site-packages (3.0.4)
Requirement already satisfied: numpy in c:\programdata\anaconda3\lib\site-
packages (from xgboost) (1.26.4)
Requirement already satisfied: scipy in c:\programdata\anaconda3\lib\site-
packages (from xgboost) (1.13.1)
```

[19]:
```
# Step 1: Define target and features
X = df.drop(columns=["salary_usd"])
y = df["salary_usd"]

# Step 2: Split categorical & numeric columns
categorical_cols = X.select_dtypes(include=["object"]).columns.tolist()
numeric_cols = X.select_dtypes(include=["int64", "float64"]).columns.tolist()

# Step 3: Preprocessor
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder, StandardScaler

preprocessor = ColumnTransformer([
    ("cat", OneHotEncoder(handle_unknown="ignore", max_categories=20),␣
 ↪categorical_cols),
    ("num", StandardScaler(), numeric_cols)
])
```

[20]:
```
from sklearn.ensemble import RandomForestRegressor
from xgboost import XGBRegressor
from sklearn.pipeline import Pipeline


model = Pipeline([
    ("preprocessor", preprocessor),
    ("regressor", RandomForestRegressor(n_estimators=200, random_state=42))
])
```

[21]:
```
from sklearn.ensemble import RandomForestRegressor
```

```
model = Pipeline([
    ("preprocessor", preprocessor),
    ("regressor", RandomForestRegressor(n_estimators=200, random_state=42))
])

model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print("R² Score:", r2_score(y_test, y_pred))
print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
```

```
R² Score: 0.8771326905597365
RMSE: 21168.85061541635
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\metrics\_regression.py:492:
FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in
1.6. To calculate the root mean squared error, use the
function'root_mean_squared_error'.
  warnings.warn(
```

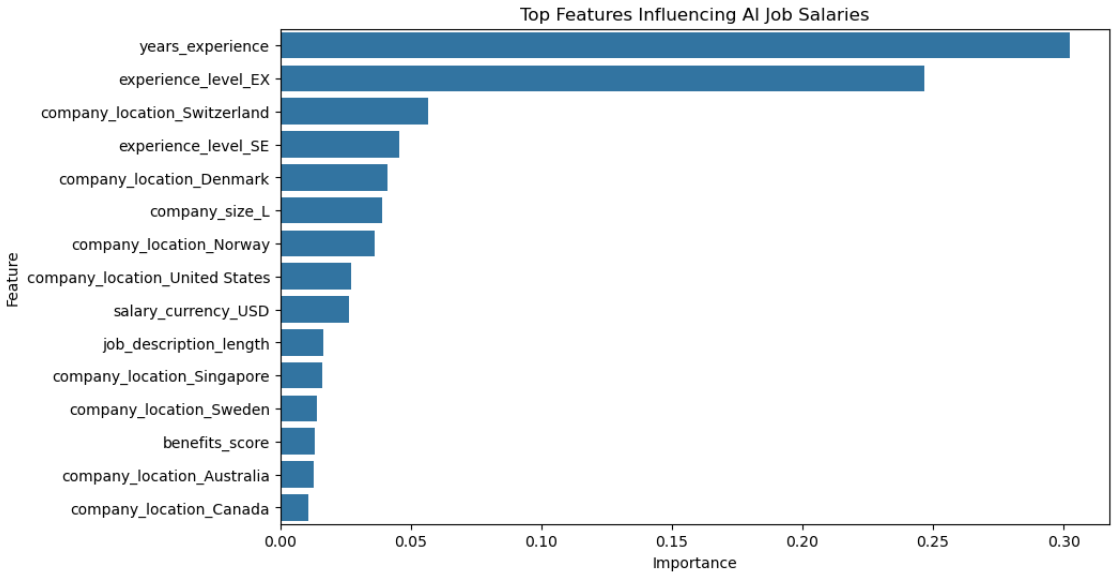
[22]:
```
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
import pandas as pd

models = {
    "Linear Regression": LinearRegression(),
    "Random Forest": RandomForestRegressor(n_estimators=200, random_state=42)
}

results = []

for name, reg in models.items():
    model = Pipeline([
        ("preprocessor", preprocessor),
        ("regressor", reg)
    ])
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)

    results.append({
        "Model": name,
        "R²": r2_score(y_test, y_pred),
        "RMSE": mean_squared_error(y_test, y_pred, squared=False)
    })

results_df = pd.DataFrame(results)
print(results_df)
```

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\metrics\_regression.py:492:
FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in
1.6. To calculate the root mean squared error, use the
function'root_mean_squared_error'.
  warnings.warn(
```

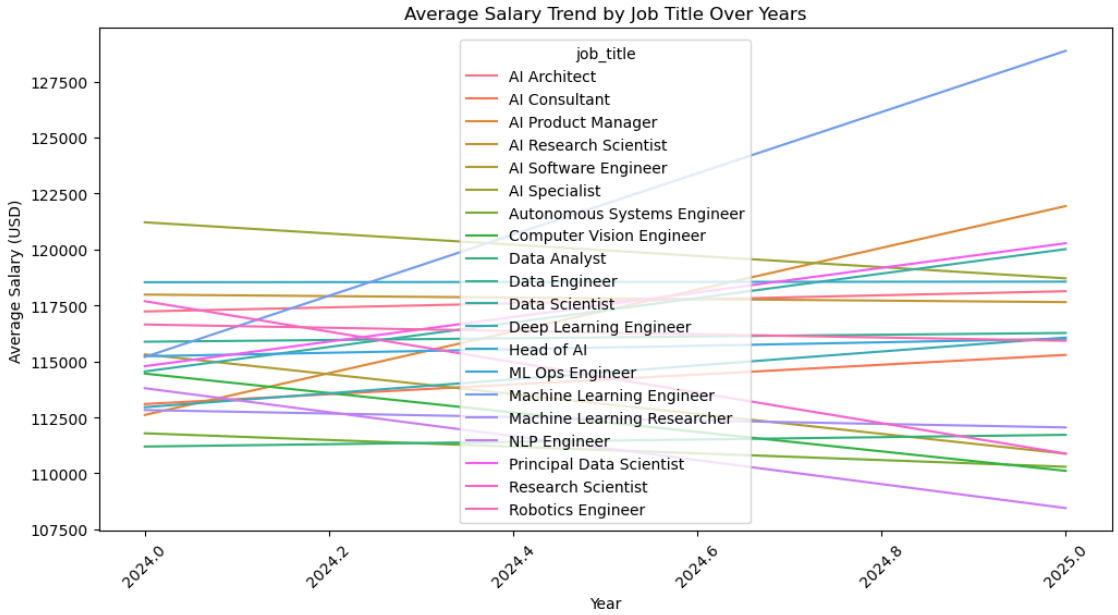
	Model	R^2	RMSE
0	Linear Regression	0.843757	23871.453606
1	Random Forest	0.877133	21168.850615

```
C:\ProgramData\anaconda3\Lib\site-packages\sklearn\metrics\_regression.py:492:
FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in
1.6. To calculate the root mean squared error, use the
function'root_mean_squared_error'.
  warnings.warn(
```

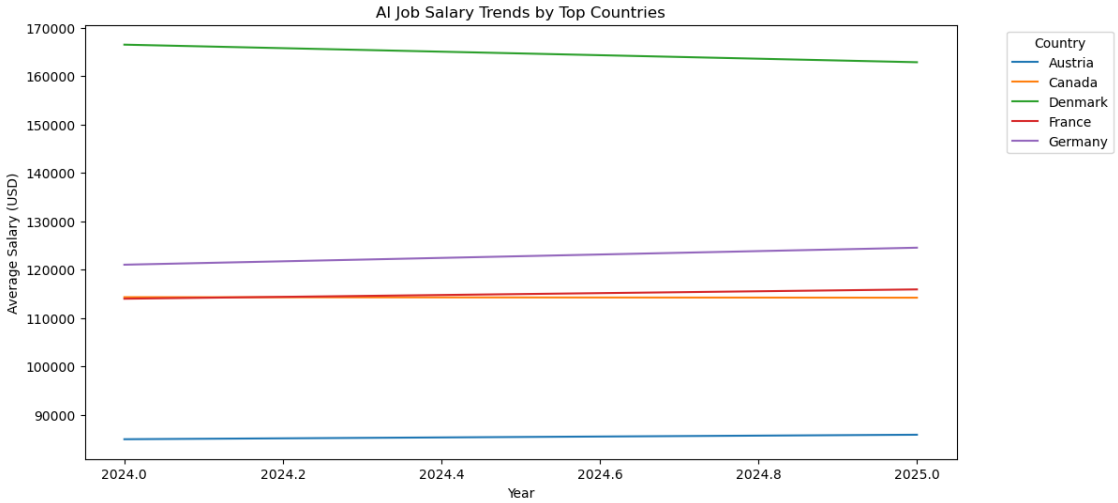
[23]:
```
best_model = RandomForestRegressor(n_estimators=200, random_state=42)
pipeline = Pipeline([
    ("preprocessor", preprocessor),
    ("regressor", best_model)
])
pipeline.fit(X_train, y_train)

# Extract feature names from one-hot encoding
encoded_features = pipeline.named_steps["preprocessor"].transformers_[0][1].
 ↪get_feature_names_out(categorical_cols)
all_features = list(encoded_features) + numeric_cols

importances = best_model.feature_importances_

feat_imp = pd.DataFrame({"Feature": all_features, "Importance": importances})
feat_imp = feat_imp.sort_values("Importance", ascending=False).head(15)

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(10,6))
sns.barplot(data=feat_imp, x="Importance", y="Feature")
plt.title("Top Features Influencing AI Job Salaries")
plt.show()
```



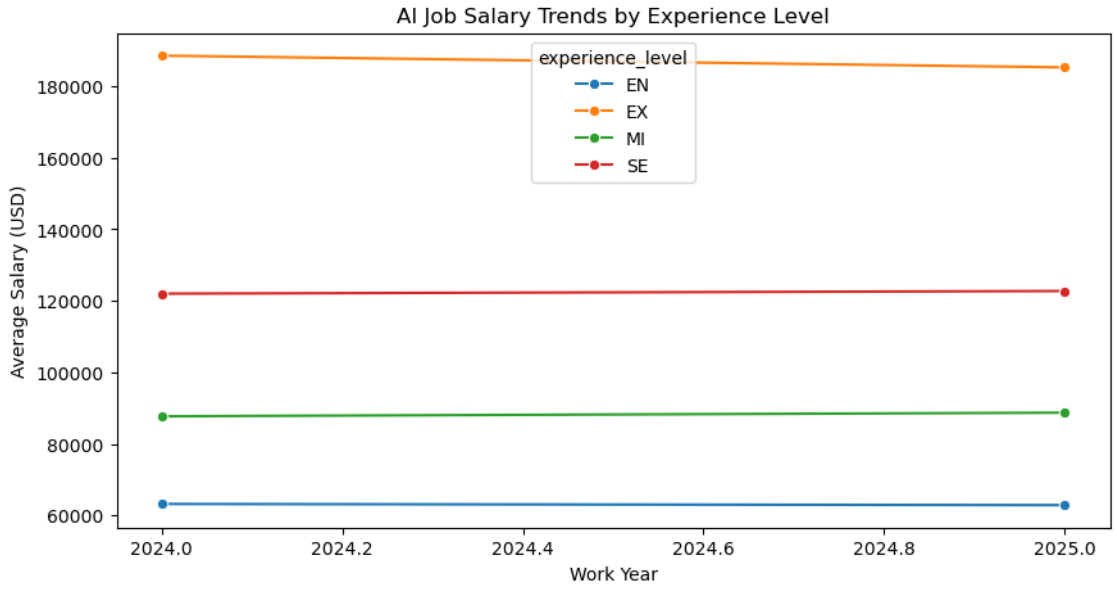
[24]:
```
role_trend = df.groupby(["year", "job_title"])["salary_usd"].mean().
 ↪reset_index()

plt.figure(figsize=(12,6))
sns.lineplot(data=role_trend, x="year", y="salary_usd", hue="job_title")
plt.title("Average Salary Trend by Job Title Over Years")
plt.xlabel("Year")
plt.ylabel("Average Salary (USD)")
plt.xticks(rotation=45)
plt.show()
```



[25]:
```
top_countries = df["company_location"].value_counts().head(5).index

loc_trend = df[df["company_location"].isin(top_countries)] \
    .groupby(["year", "company_location"])["salary_usd"].mean().reset_index()

plt.figure(figsize=(12,6))
sns.lineplot(data=loc_trend, x="year", y="salary_usd", hue="company_location")
plt.title("AI Job Salary Trends by Top Countries")
plt.ylabel("Average Salary (USD)")
plt.xlabel("Year")
plt.legend(title="Country", bbox_to_anchor=(1.05, 1), loc="upper left")
plt.show()
```



[26]:
```
exp_trend = df.groupby(["year", "experience_level"])["salary_usd"].mean().
 ↪reset_index()

plt.figure(figsize=(10,5))
sns.lineplot(data=exp_trend, x="year", y="salary_usd", hue="experience_level",␣
 ↪marker="o")
plt.title("AI Job Salary Trends by Experience Level")
plt.ylabel("Average Salary (USD)")
plt.xlabel("Work Year")
plt.show()
```

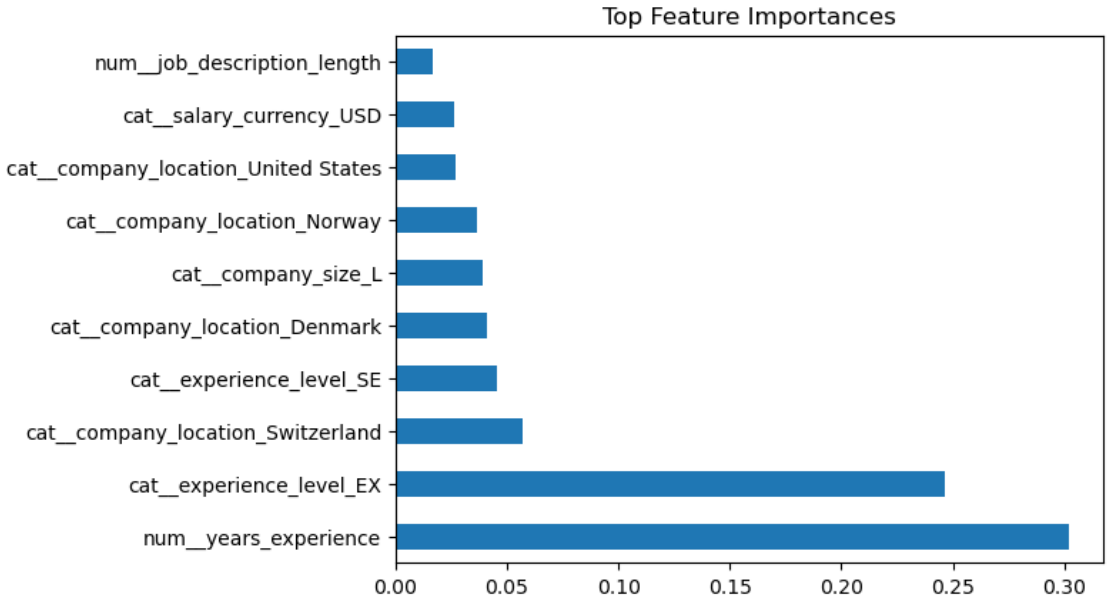


[27]:
```
import pandas as pd

feature_names = model.named_steps["preprocessor"].get_feature_names_out()
importances = model.named_steps["regressor"].feature_importances_

fi = pd.DataFrame({"Feature": feature_names, "Importance": importances})
fi = fi.sort_values(by="Importance", ascending=False)
print(fi.head(20))
```

```
                              Feature  Importance
170               num__years_experience    0.302338
44              cat__experience_level_EX    0.246742
67     cat__company_location_Switzerland    0.056756
46              cat__experience_level_SE    0.045387
```

```
55          cat__company_location_Denmark    0.040924
71                    cat__company_size_L    0.038983
63           cat__company_location_Norway    0.036193
69   cat__company_location_United States    0.027081
42               cat__salary_currency_USD    0.026321
171           num__job_description_length    0.016366
64         cat__company_location_Singapore    0.016233
66            cat__company_location_Sweden    0.014056
172                   num__benefits_score    0.013231
51         cat__company_location_Australia    0.012963
53            cat__company_location_Canada    0.010916
43                cat__experience_level_EN    0.006925
45                cat__experience_level_MI    0.006762
73                     cat__company_size_S    0.006347
169                      num__remote_ratio    0.003797
72                     cat__company_size_M    0.003243
```

[28]:
```
importances = model.named_steps['regressor'].feature_importances_
feature_names = preprocessor.get_feature_names_out()
feat_imp = pd.Series(importances, index=feature_names).
 ↪sort_values(ascending=False)
feat_imp.head(10).plot(kind='barh')
plt.title("Top Feature Importances")
plt.show()
```



[30]:
```
import joblib
joblib.dump(model, "salary_prediction_model.pkl")
```

[30]:
```
['salary_prediction_model.pkl']
```

[43]:
```python
def build_input(df, values: dict):
    row = pd.DataFrame([values])
    # enforce same dtypes as training df
    for col in df.columns:
        if col in row.columns:
            try:
                row[col] = row[col].astype(df[col].dtype, errors="ignore")
            except Exception:
                pass  # if conversion fails, keep original
    return row
```

[44]:
```python
import joblib
loaded_model = joblib.load("salary_prediction_model.pkl")

new_data = build_input(df, {
    "job_id": "101",
    "job_title": "Machine Learning Engineer",
    "company_name": "OpenAI",
    "company_location": "US",
    "employee_residence": "US",
    "industry": "Tech",
    "employment_type": "FT",
    "work_setting": "Remote",
    "experience_level": "SE",
    "company_size": "M",
    "education_required": "Bachelor",
    "required_skills": "Python, Machine Learning, Deep Learning",
    "years_experience": 3,
    "job_description_length": 500,
    "benefits_score": 0.8,
    "salary_currency": "USD",
    "remote_ratio": 100,
    "posting_date": pd.to_datetime("2025-01-01"),
    "application_deadline": "2025-12-31",
    "year": 2025
})



predicted_salary = loaded_model.predict(new_data)
print("Predicted Salary (USD):", predicted_salary)
```

```
Predicted Salary (USD): [73345.055]
```

[45]:
```
from sklearn.metrics import r2_score, mean_squared_error

y_pred = model.predict(X)
print("R² Score:", r2_score(y, y_pred))
print("RMSE:", mean_squared_error(y, y_pred, squared=False))
```

```
R² Score: 0.961839241823746
RMSE: 11771.44900069712

C:\ProgramData\anaconda3\Lib\site-packages\sklearn\metrics\_regression.py:492:
FutureWarning: 'squared' is deprecated in version 1.4 and will be removed in
1.6. To calculate the root mean squared error, use the
function'root_mean_squared_error'.
  warnings.warn(
```

[ ]: