Fantastic Four

Data Science Salaries

Data Science Salaries Project Overview

Project Goal: Create a tool for students to input job type and experience level to find corresponding salaries

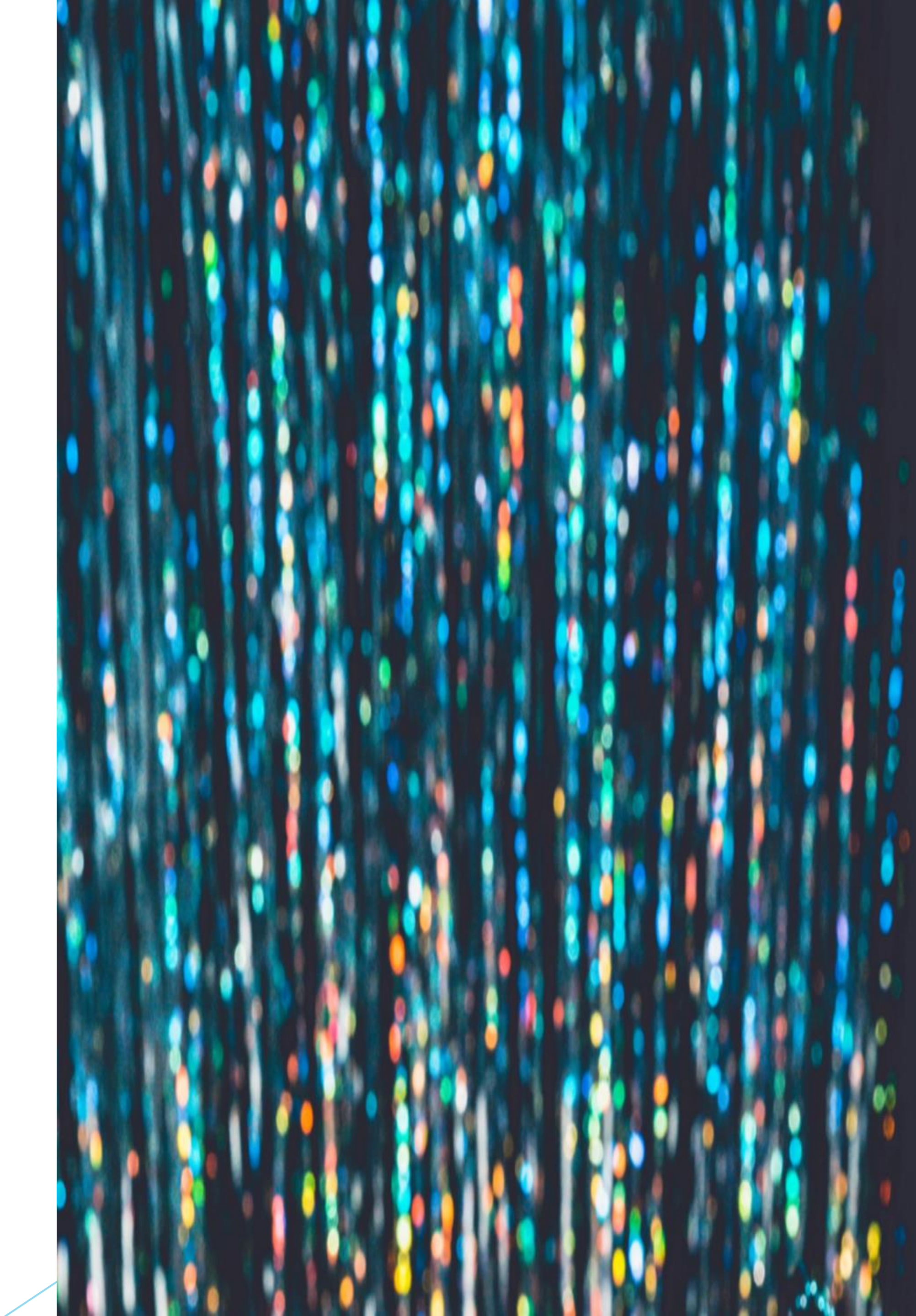
Data Source: Data Science Salaries 2024 from Kaggle, which includes salaries from 75 countries.

Average Salaries:

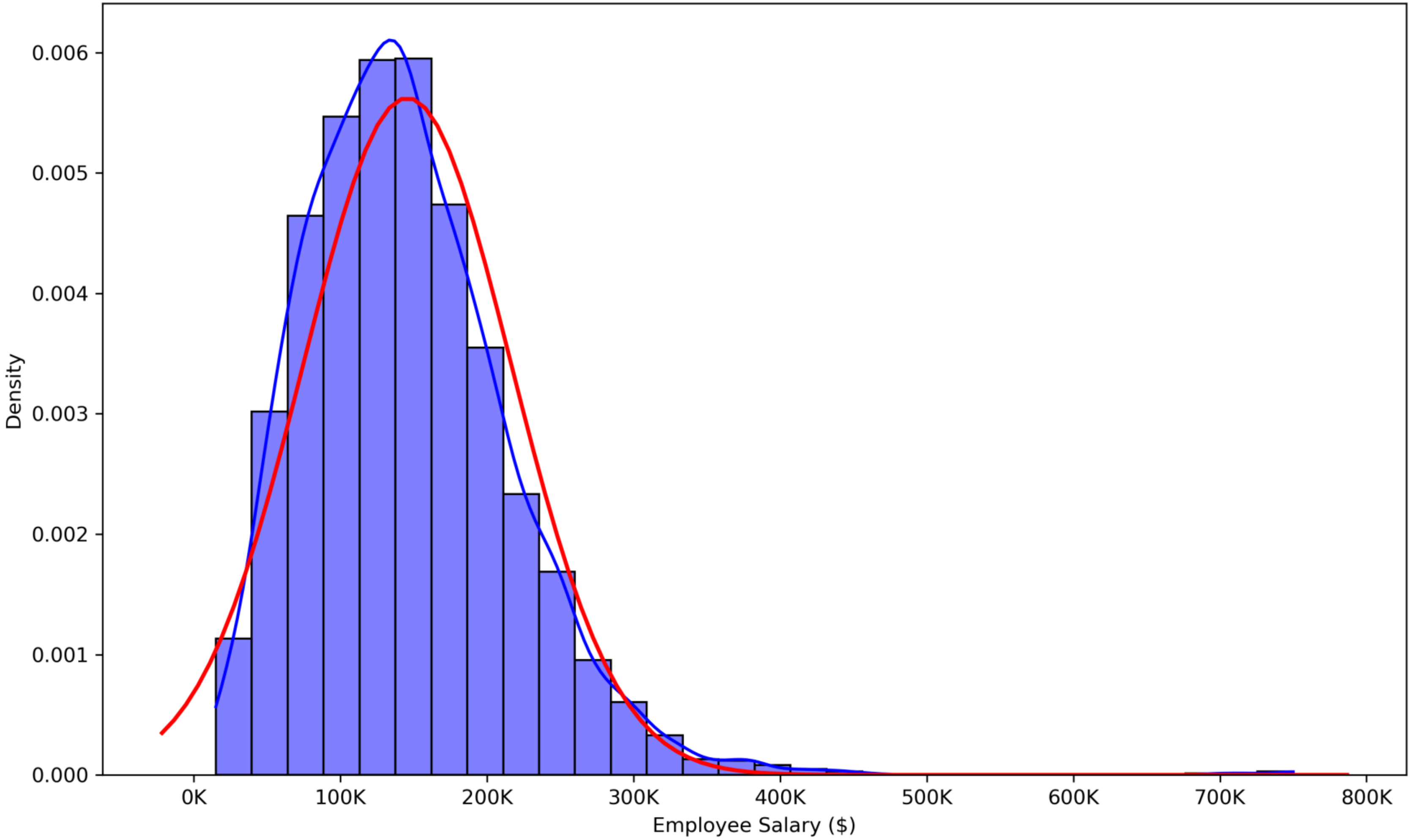
- Entry Level - $85,000 annually
- Mid-level - $120,000 annually
- Senior level - $162,000 annually
- Executive level - $190,000 annually

Machine Learning Techniques:

- Multi-class classification
- Ada Boost
- Linear regression
- Unsupervised learning



2020-2024 Employee Salary Distribution with Normal Curve



Classification Models

Models to predict salary classes based on certain features:

Features

1. job title
2. experience level
3. employment type
4. work model
5. employee residence
6. company location
7. company size

Classifiers

1. K Nearest Neighbors
2. Support Vector
3. Decision Tree
4. Random Forest
5. AdaBoost

Bins:

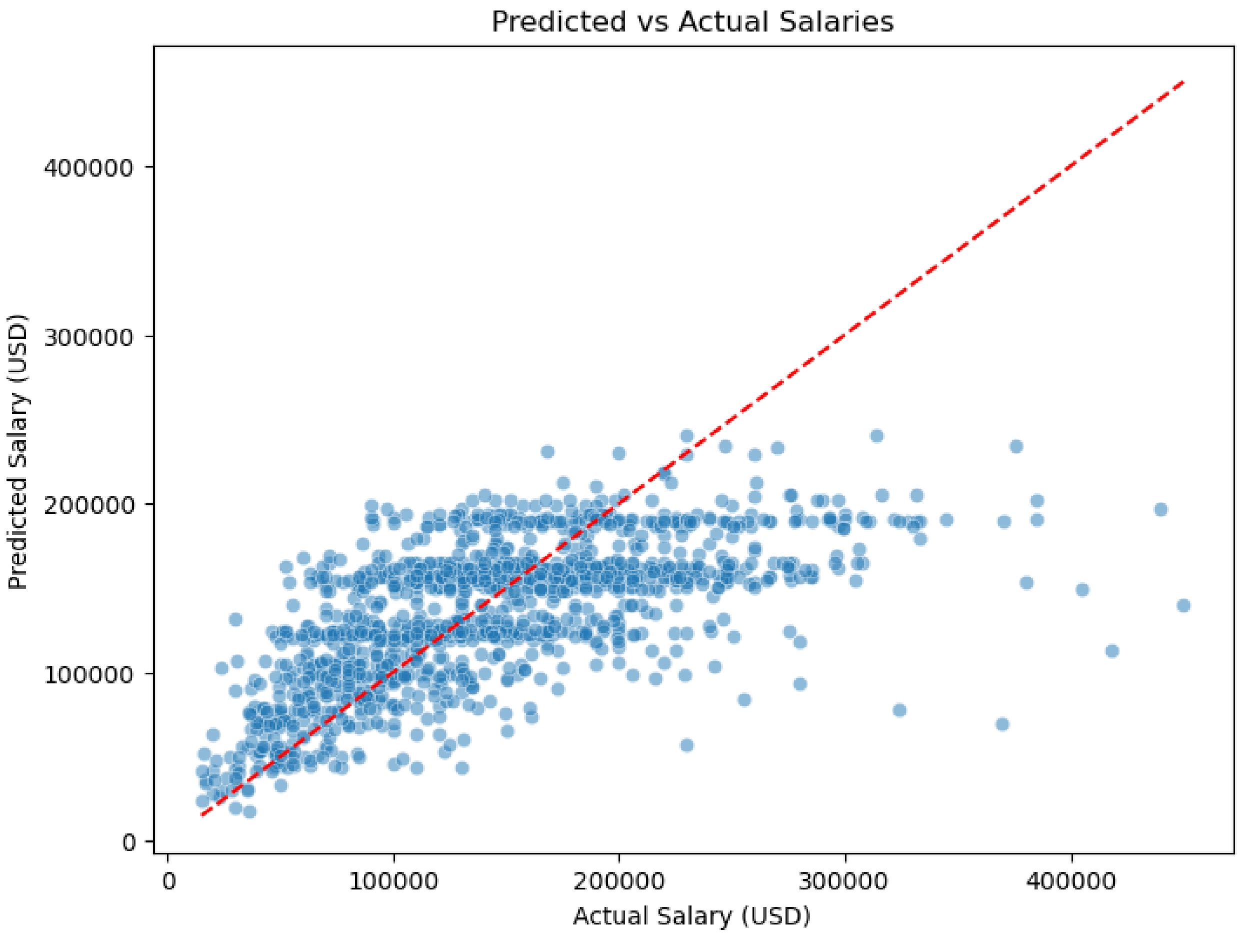
Class	Bins	Salary Range	Count
0	very low	1,500 – 50,000	383
1	low	50,001 – 156,000	3651
2	average	156,001 – 176,000	652
3	high	176,001 – 750,000	1913

AdaBoost (63%) Classification Report:

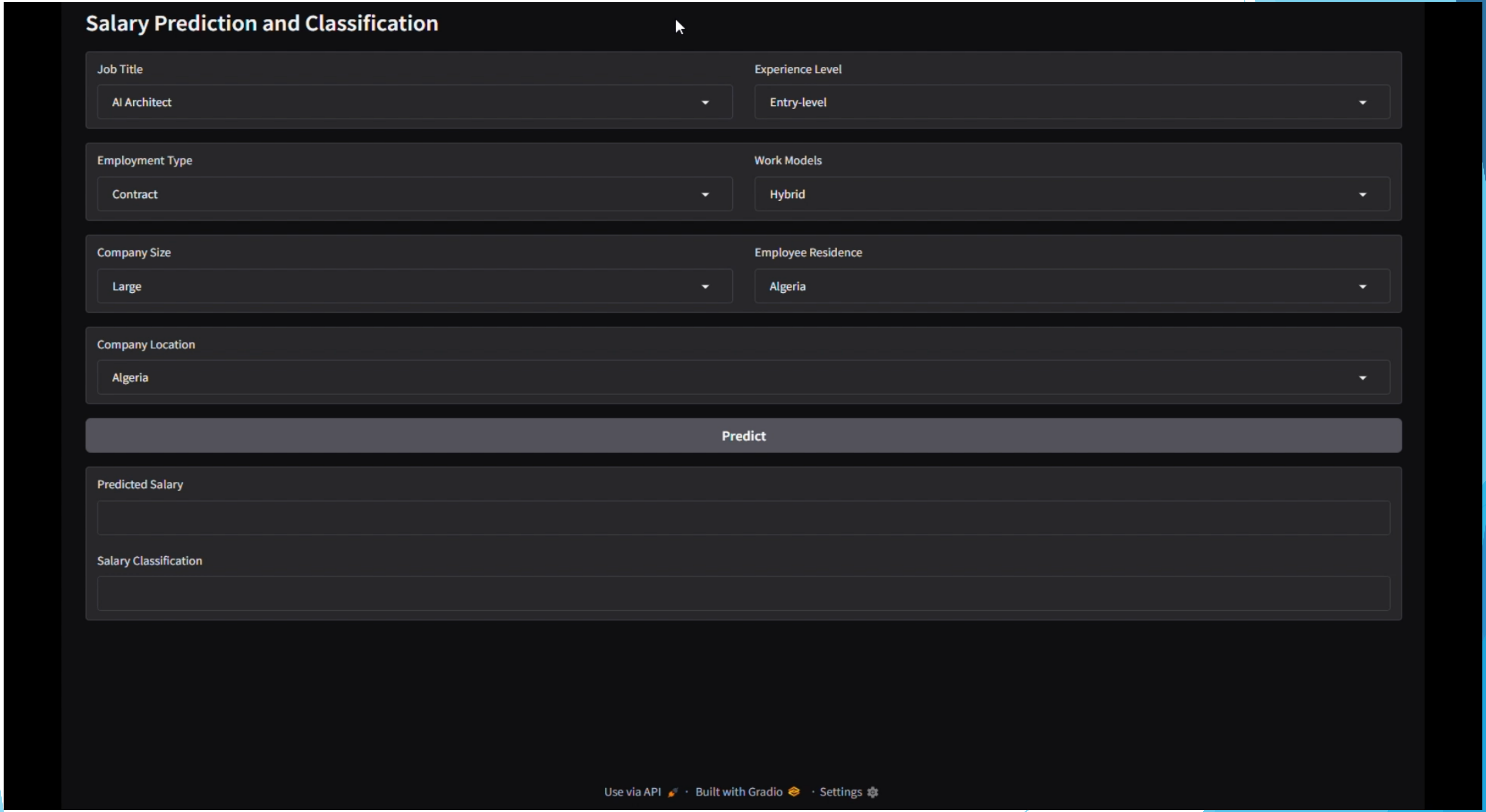
Class	Precision	Recall	F1-Score	Support
0	0.50	0.01	0.01	165
1	0.53	0.49	0.51	448
2	0.67	0.83	0.74	945
3	0.64	0.41	0.50	92
Accuracy			0.63	1650
Macro Avg	0.59	0.43	0.44	1650
Weighted Avg	0.61	0.63	0.59	1650

Linear Regression Model to Predict a Salary

- **Data Preprocessing:** Removed outliers, applied log transformation, one-hot encoded categorical features, and standardized numerical data.
- **Model:** Used **Ridge Regression** to reduce overfitting and handle multicollinearity.
- **Performance: Test R² = 0.5211**, **Test MSE = 0.1434**.
- **Key Takeaway:** Interactive tool for real-time salary prediction.



Gradio Presentation



Salary Prediction and Classification

Job Title: AI Architect

Experience Level: Entry-level

Employment Type: Contract

Work Models: Hybrid

Company Size: Large

Employee Residence: Algeria

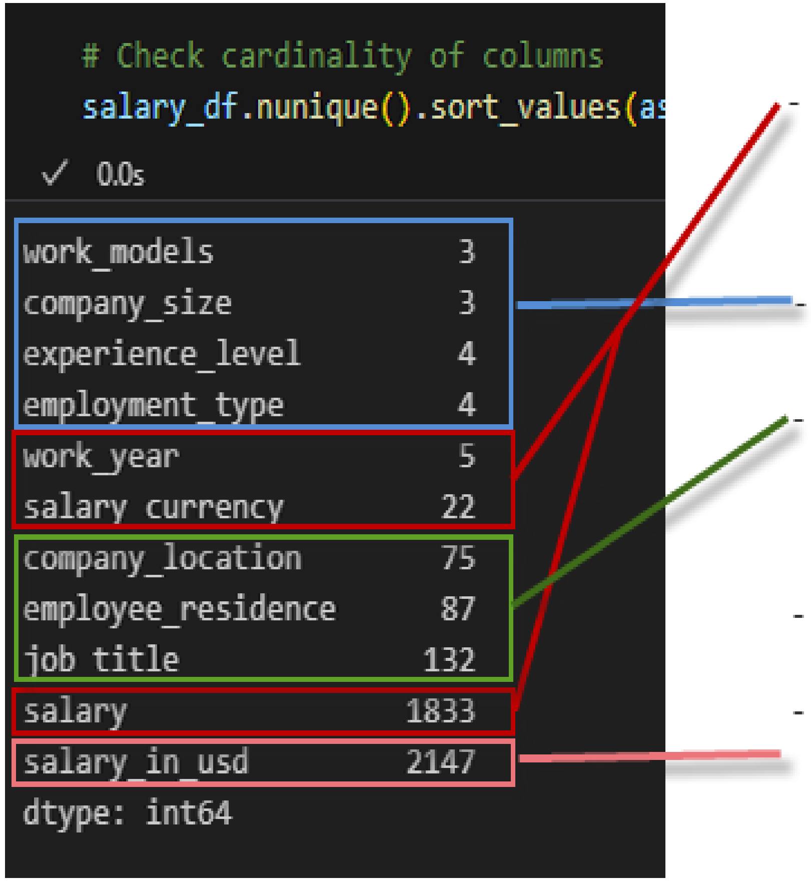
Company Location: Algeria

Unsupervised Learning Analysis

Why?

- Relatively low metrics from supervised learning models.
- What are the important features?
- How would the data cluster?

Preprocessing



```
# Check cardinality of columns
salary_df.nunique().sort_values(as
```

```
✓ 0.0s
```

```
work_models             3
company_size            3
experience_level        4
employment_type         4
work_year               5
salary_currency        22
company_location       75
employee_residence     87
job_title             132
salary               1833
salary_in_usd        2147
dtype: int64
```

- Drop irrelevant columns

- Encode **low** cardinality columns with *get_dummies*

- Encode **high** cardinality columns with *frequency encoder*

- Only remaining numeric column: salary_in_usd

- Remove 107 outliers beyond upper_bound
  lower_bound = Q1 - 1.5 * IQR = - 40,000
  upper_bound = Q3 + 1.5 * IQR = 320,000

- Scale with StandardScaler()

Unsupervised Learning Analysis

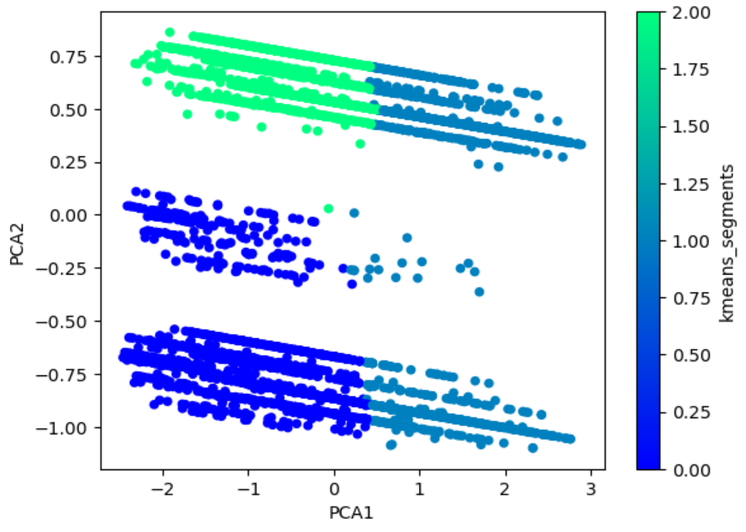
Clustering PCA Optimized Data (K=3)

- 3-component Variance Ratio : [0.48, 0.21, 0.14] *sum = 0.83*

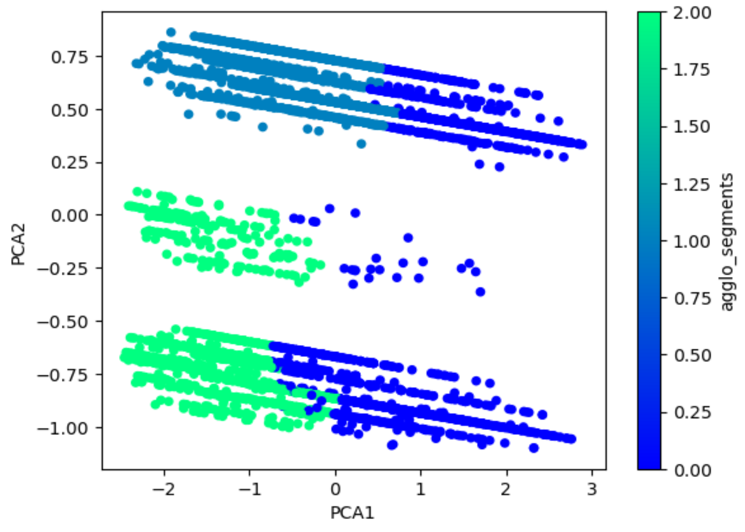
PCA1	salary_in_usd	wt:	0.936
PCA2	work_models_On-site	wt:	0.697
PCA3	experience_level_Senior	wt:	0.699

- Surprising light weights: job_title, employee_residency and company_location

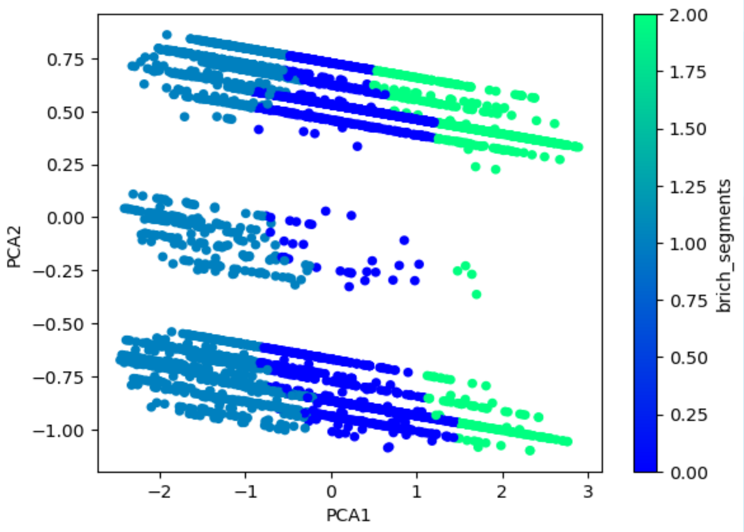
K-Means



Agglomerative



Birch



Conclusion:

Classification

- Predicted and classified salaries into buckets ranging from very low to high

Linear Regression

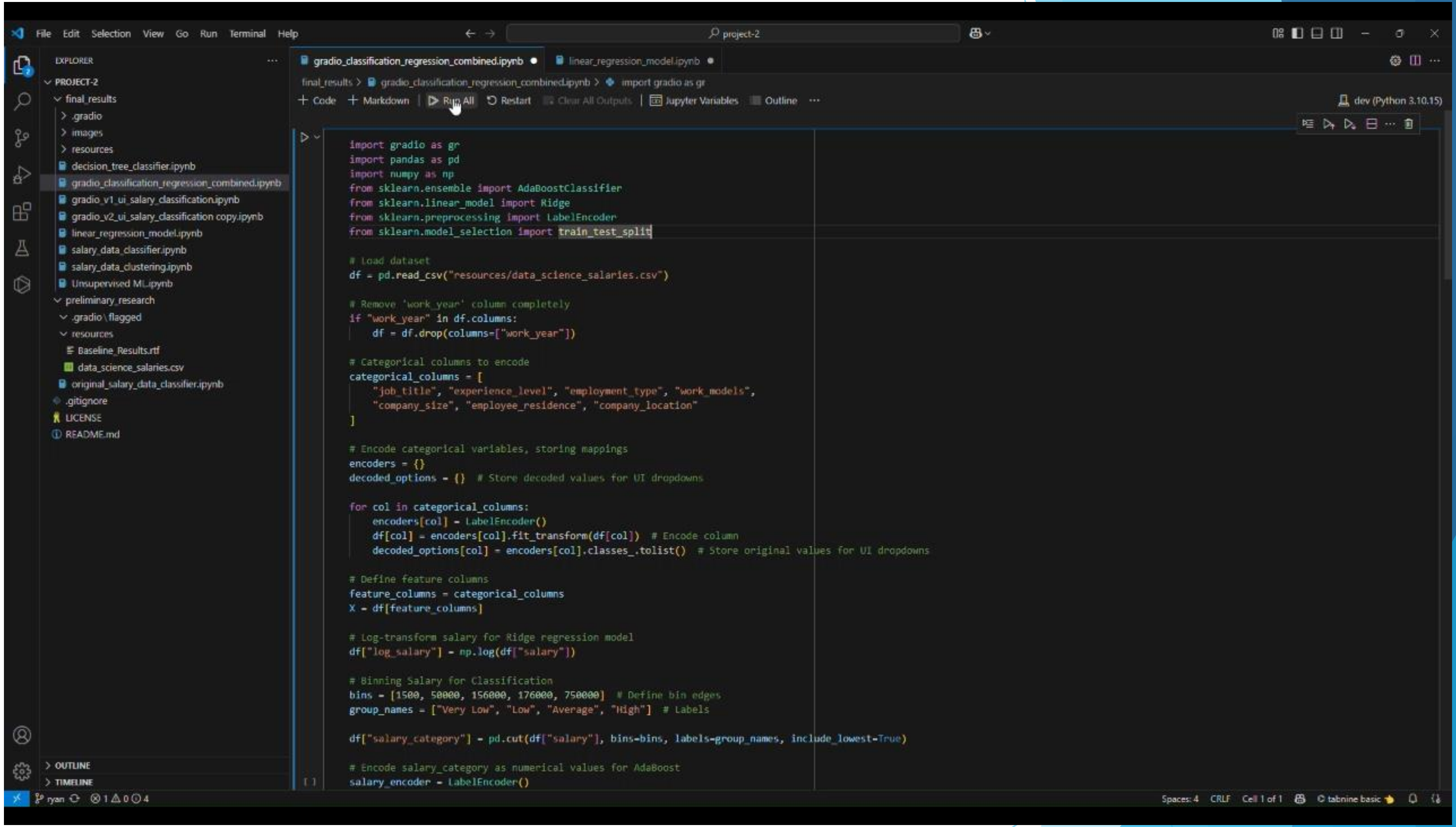
- Ridge Regression allowed us to predict the actual salary for a set of job specific features

Unsupervised Learning

- Gave insights into the key features such as work model and experience level were found to be the most impactful features

Low accuracy and R scores

- High variability in salaries within job title lower model accuracy.
- The R2 scores were likely affected by missing factors such as education level


File
Edit
Selection
View
Go
Run
Terminal
Help
project-2
EXPLORER
PROJECT-2
final_results
.gradio
images
resources
decision_tree_classifier.ipynb
gradio_classification_regression_combined.ipynb
gradio_v1_ui_salary_classification.ipynb
gradio_v2_ui_salary_classification copy.ipynb
linear_regression_model.ipynb
salary_data_classifier.ipynb
salary_data_clustering.ipynb
Unsupervised ML.ipynb
preliminary_research
.gradio \ flagged
resources
Baseline_Results.rtf
data_science_salaries.csv
original_salary_data_classifier.ipynb
.gitignore
LICENSE
README.md
OUTLINE
TIMELINE
gradio_classification_regression_combined.ipynb
linear_regression_model.ipynb
final_results > gradio_classification_regression_combined.ipynb > import gradio as gr
+ Code
+ Markdown
Run All
Restart
Clear All Outputs
Jupyter Variables
Outline
dev (Python 3.10.15)
ryan
Spaces: 4
CRLF
Cell 1 of 1
tabnine basic

```python
import gradio as gr
import pandas as pd
import numpy as np
from sklearn.ensemble import AdaBoostClassifier
from sklearn.linear_model import Ridge
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

# Load dataset
df = pd.read_csv("resources/data_science_salaries.csv")

# Remove 'work_year' column completely
if "work_year" in df.columns:
    df = df.drop(columns=["work_year"])

# Categorical columns to encode
categorical_columns = [
    "job_title", "experience_level", "employment_type", "work_models",
    "company_size", "employee_residence", "company_location"
]

# Encode categorical variables, storing mappings
encoders = {}
decoded_options = {}  # Store decoded values for UI dropdowns

for col in categorical_columns:
    encoders[col] = LabelEncoder()
    df[col] = encoders[col].fit_transform(df[col])  # Encode column
    decoded_options[col] = encoders[col].classes_.tolist()  # Store original values for UI dropdowns

# Define feature columns
feature_columns = categorical_columns
X = df[feature_columns]

# Log-transform salary for Ridge regression model
df["log_salary"] = np.log(df["salary"])

# Binning Salary for Classification
bins = [1500, 50000, 156000, 176000, 750000]  # Define bin edges
group_names = ["Very Low", "Low", "Average", "High"]  # Labels

df["salary_category"] = pd.cut(df["salary"], bins=bins, labels=group_names, include_lowest=True)

# Encode salary_category as numerical values for AdaBoost
salary_encoder = LabelEncoder()
```

Questions

Link: https://github.com/tlockhart/project-2