Module 05: Lab 02

Regression Modeling on Employment Data

Yanyang He

April 14, 2025

Objectives

1. Use **PySpark** to process the Lightcast dataset.
2. Engineer features from structured columns for salary prediction.
3. Train **Linear Regression model**.
4. Evaluate models using **RMSE** and **R^2**.
5. Visualize predictions using diagnostic plots.
6. Push work to GitHub and submit the repository link.

Setup

The instruction below provides you with general keywords for columns used in the lightcast file. See the data schema generated after the load dataset code above to use proper column name. For visualizations, tables, or summaries, please **customize colors, fonts, and styles** as appropriate to avoid a **2.5-point deduction**. Also, **provide a two-sentence explanation** describing key insights drawn from each section's code and outputs.

1. Follow the steps below as necessary, use your best judgement in importing/installing/creating/saving files as needed.
2. Create a new Jupyter Notebook in your `ad688-sp25-lab08` directory named `lab08_yourname.ipynb`, if the file exists make sure to change the name.
3. Use your **EC2 instance** for this lab.
4. Ensure the `lightcast_data.csv` file is available on the EC2 instance. if not then **Download the dataset**
5. **Add the dataset to `.gitignore`** to avoid pushing large files to GitHub. Open your `.gitignore` file and add:
6. Make sure to create a virtual environment and install the required Python libraries if needed, don't forget to activate it:

7. Install the required Python libraries if needed, you can also use the given requirement file to install the packages to the virtual environment:

```
python3 -m venv .venv
source .venv/bin/activate
gdown https://drive.google.com/uc?id=1V2GCHGt2dkFGqVBeoUFckU4IhUgk4ocQ
echo "lightcast_job_postings.csv" >> .gitignore
pip install -r requirements.txt
```

1 Load the Dataset

1. **Load the Raw Dataset**:
    - Use Pyspark to the `lightcast_data.csv` file into a DataFrame:
    - You can reuse the previous code.
    - Copying code from your friend constitutes plagiarism. DO NOT DO THIS.

```
from pyspark.sql import SparkSession
import pandas as pd
import plotly.express as px
import plotly.io as pio
pio.renderers.default = "notebook"

# Initialize Spark Session
spark = SparkSession.builder.appName("LightcastData").getOrCreate()
spark.conf.set('spark.sql.repl.eagerEval.enabled', True)
spark.conf.set('spark.sql.repl.eagerEval.maxNumRows', 20)

# Load Data
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","t
```

```
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
25/04/14 20:33:57 WARN NativeCodeLoader: Unable to load native-hadoop library for your platfo
```

2 Feature Engineering

Feature Engineering is a crucial step in preparing your data for machine learning. In this lab, we will focus on the following tasks:

1. **Drop rows with missing values** in the target variable and key features.
2. By now you are already familiar with the code and the data. Based on your understanding please choose any 3 (my code output has 10) variables as:
    1. two continuous variables (use your best judgment!)
    2. one categorical.
    3. Your dependent variable (y) is `SALARY`.
3. **Convert categorical variables** into numerical representations using **StringIndexer** and **OneHotEncoder**.
4. **Assemble features** into a single vector using **VectorAssembler**.
5. **Split the data** into training and testing sets.

```
from pyspark.sql.functions import col
from pyspark.ml.feature import StringIndexer, OneHotEncoder, VectorAssembler
from pyspark.ml import Pipeline

df = df.dropna(subset=[
    "SALARY",
    "REMOTE_TYPE_NAME",
    "DURATION",
    "MIN_YEARS_EXPERIENCE",
])
# from previous labs, we know that average salary of remote/on site/hybrid are quite differe
categorical_cols = ["MIN_EDULEVELS_NAME"]
# we only have two numerical variables: Duration and YOE
continuous_cols = ["DURATION", "MIN_YEARS_EXPERIENCE"]

indexers = [StringIndexer(inputCol=col, outputCol=f"{col}_idx", handleInvalid='skip') for co
encoders = [OneHotEncoder(inputCol=f"{col}_idx", outputCol=f"{col}_vec") for col in categori

assembler = VectorAssembler(
    inputCols=continuous_cols + [f"{col}_vec" for col in categorical_cols],
    outputCol="features",
)

pipeline = Pipeline(stages=indexers + encoders + [assembler])
data = pipeline.fit(df).transform(df).select("SALARY", "features")
data.show(5,False)
```

```
+------+--------------------------+
|SALARY|features                  |
+------+--------------------------+
|192800|(7,[0,1,2],[55.0,6.0,1.0]) |
|125900|(7,[0,1,5],[18.0,12.0,1.0])|
|118560|(7,[0,1,3],[20.0,5.0,1.0]) |
|192800|(7,[0,1,2],[55.0,6.0,1.0]) |
|116500|(7,[0,1,5],[16.0,12.0,1.0])|
+------+--------------------------+
only showing top 5 rows
```

3 Train/Test Split

- Perform a **random split** of the data into training and testing sets.
- Set a random seed for reproducibility.
- You can choose a number for splitting to your liking, justify your choice.

```
14416
```

```
                                                                [Stage 12:>
```

```
(10217, 2)
(4199, 2)
```

4 Linear Regression

- Train a **Linear Regression** model using the training data. You will run in to an important issue here. Please make an effort in figuring it by yourself. This is one of the most asked interview questions in CapitalOne's management recruiting program.
- Evaluate the model on the test data.
- Print the coefficients, intercept, R^2, RMSE, and MAE.
- Use the `summary` object to extract the coefficients and their standard errors, t-values, and p-values.

- Create a DataFrame to display the coefficients, standard errors, t-values, p-values, and confidence intervals.
- Interpret the coefficients and their significance and explain the model performance metrics.

```
25/04/14 20:34:23 WARN Instrumentation: [9f32da9e] regParam is zero, which might cause numeri
25/04/14 20:34:27 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netli
25/04/14 20:34:27 WARN InstanceBuilder: Failed to load implementation from:dev.ludovic.netli
```

	r2	rmse	mae
0	0.375663	33844.806259	26245.314927

Interpretaion of model summaries:

- R-square 0.376: only 37% variation in salries can be explained by the model
- RMSE and MAE: very large, the model cannot give precise predictions

We have only take 3 variables as input, many other variables may play have significant influence on salaries.

	rmse	r2	mae
0	33134.004218	0.369048	25758.992898

Interpreations of results on test set:

- RMSE/MAE/R-Square are similar to the result on training set
- our model can genearlize to unseen data
- it has a consistent performance on train/test set

However, as we said before, the model cannot give a precise prediction and can only explain for 37% of the variation in the salary

	Intercept and Coefficients	Std Error	t-stat	P-Value
Intercept	98812.681381	23.372127	-0.313938	7.535746e-01
Feature 1	-7.337397	106.549389	70.225843	0.000000e+00

	Intercept and Coefficients	Std Error	t-stat	P-Value
Feature 2	7482.520653	8222.987333	-2.011999	4.424626e-02
Feature 3	-16544.642263	8257.008684	-1.785402	7.422586e-02
Feature 4	-14742.077469	8305.820642	-5.765106	8.397403e-09
Feature 5	-47883.936060	8337.317900	-5.931231	3.104303e-09
Feature 6	-49450.554328	8494.582812	3.285598	1.021094e-03
Feature 7	27909.782503	8239.300243	11.992849	0.000000e+00

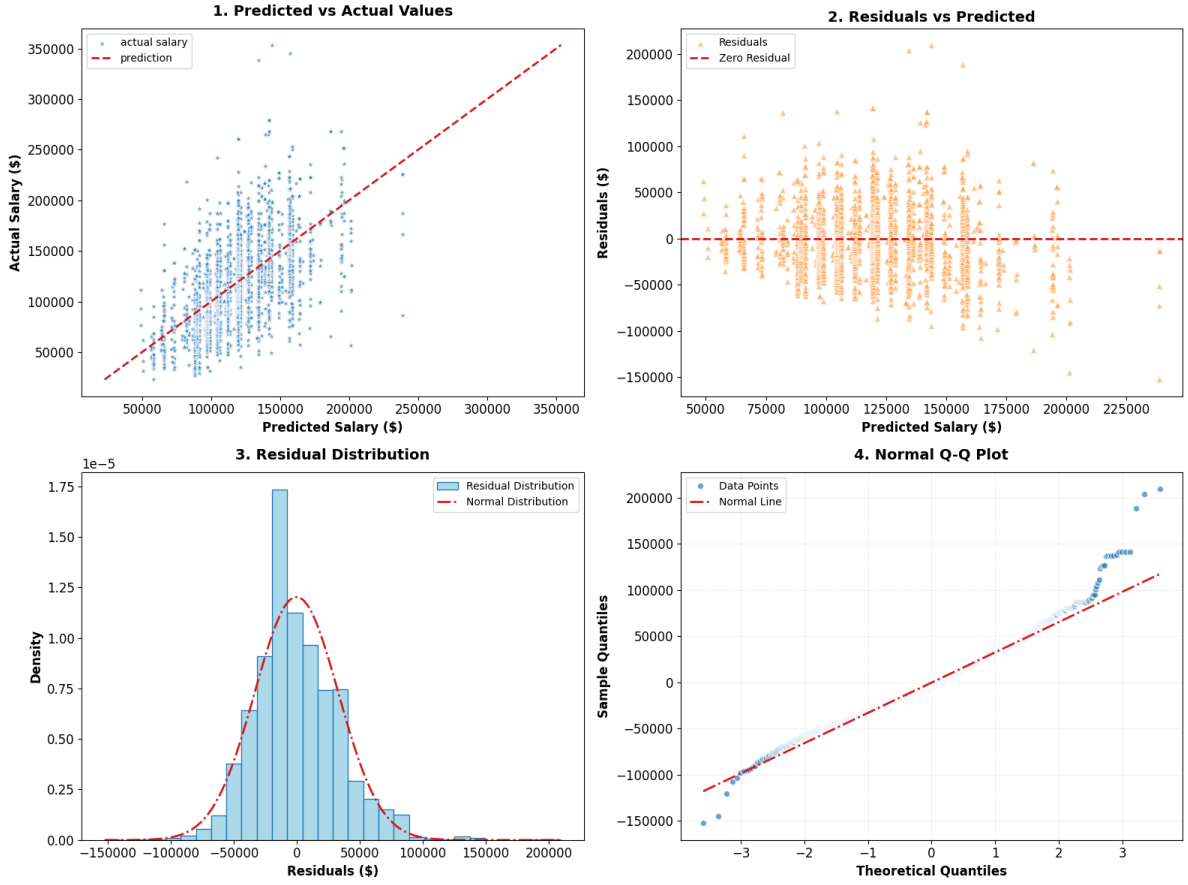
GLR Summary Interpretations:

- Intercept:
    1. 98.8K is the base salary level when no information is provided.
    2. Small std-err means this result is reliable.
    3. However, this result is not the p-value 0.75 is too large, so it is not statistically significant.
- coefficient for duration:
    1. Salary is negatively correlated to the job post available duration.
    2. This means jobs that high paying jobs are quickly filled on the job market. While low paying jobs remains in the market for longer time.
    3. The absolute value 7.3 is very small, meaning one more day open leads to only 7.3 reduction in the salary.
    4. low standard error, the result is reliable.
    5. $p < 0.001$ highly statistical significance
- coefficient for years of experience:
    1. The coefficient is positive, so it is positively correlated to salary. This means the more experience one have, the higher salary they get.
    2. On average, one more year of experience leads to an increase of 7.5k in the salary.
    3. low standard error, the result is reliable.
    4. $p < 0.001$ highly statistical significance.
- coefficient for remote: (coefficients for 5 binary variables)
    1. On-site jobs are likely to have higher salaries (feature 7).
    2. The coefficient absolute values are very large, meaning that remote work type (remote/hybrid/onsite) greatly influences the salary.
    3. The standard errors are very large, meaning the results are mot reliable.
    4. all of the coefficients have $p<0.001$ so they are statistically significant.

5 Diagnostic Plot

Diagnostic plots are essential for evaluating the performance of regression models. In this section, we will create several diagnostic plots to assess the linear regression model's assumptions and performance. There are four (2*2 grid) main plots we will create, you can use `seaborn` or `matplotlib` for this:

1. **Predicted vs Actual Plot**
2. **Residuals vs Predicted Plot**
3. **Histogram of Residuals**
4. **QQ Plot of Residuals**



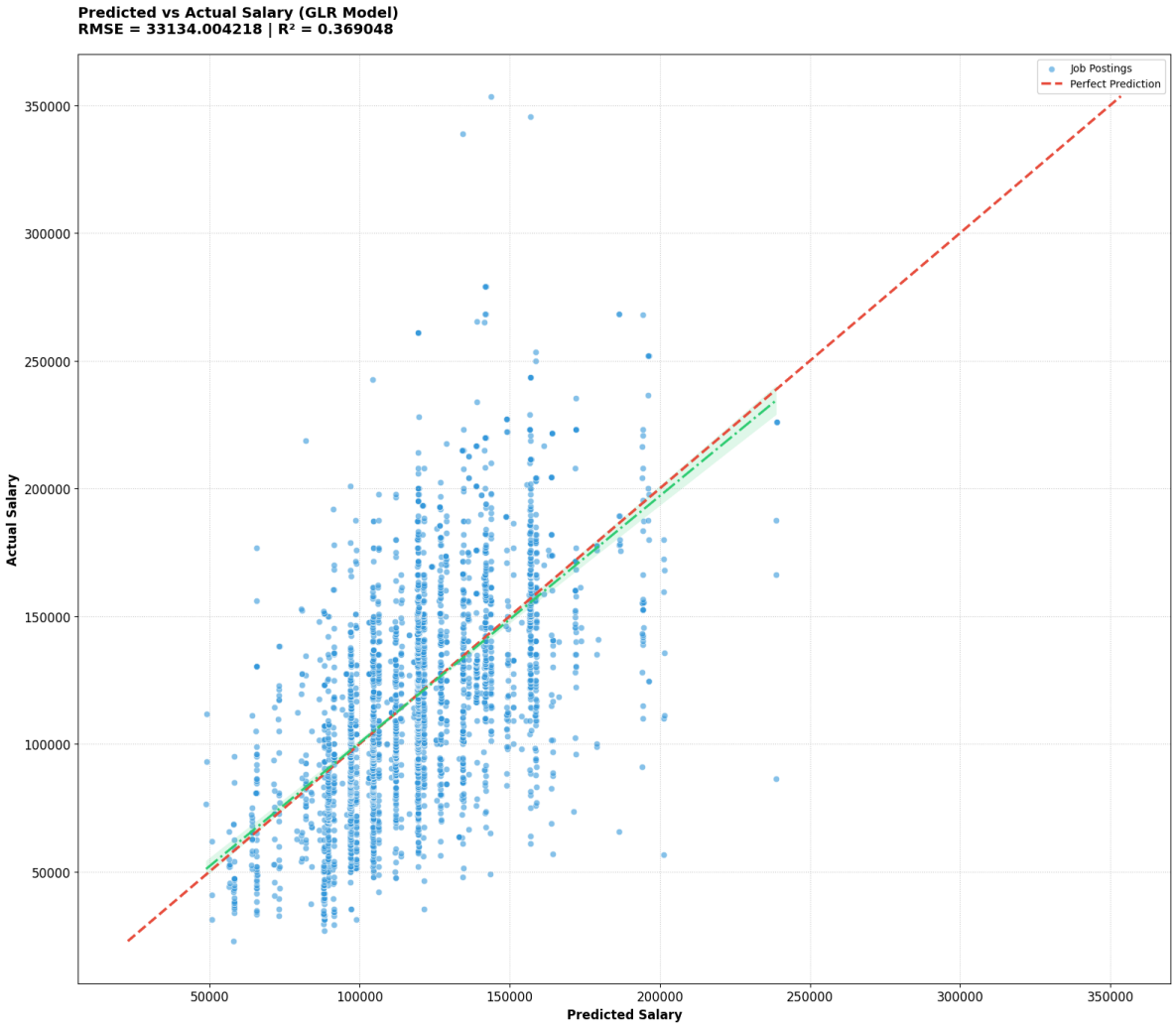
6 Evaluation

The evaluation of the model is crucial to understand its performance. In this section, we will calculate and visualize the following metrics: 1. **R^2 (Coefficient of Determination)**: Indicates how well the model explains the variance in the target variable. 2. **RMSE (Root Mean Squared Error)**: Measures the average magnitude of the errors between predicted and actual values.

	R-Square: Coefficient of Determination	RMSE: Root Mean Square Error
0	0.369048	33134.004218

6.1 Model Evaluation Plot

- Display the predicted vs actual salary plot with a red line indicating the ideal fit (y=x).
- Use `seaborn` or `matplotlib` to create the plot.
- Customize the plot with appropriate titles, labels, and legends.
- Describe the plot in a few sentences, highlighting key insights and observations.



Submission

1. Save figures in the `_output/` folder.
2. Commit and push code and output files:

```
git add .
git commit -m "Add Lab 08 Salary Prediction models and output"
git push origin main
```

3. Submit your GitHub repository link.

Resources

- PySpark MLlib Docs
- Seaborn Docs
- Pandas User Guide