Exam 3 Take-Home: Annual Salaries of College Grads

Lewis Collum

Updated: April 27, 2020

Introduction

We are given data containing the annual salaries of participants that have graduated college. Along with the salaries, we have features including, university type (public, private), gender, years since graduation, and cost of education. We administer a Multi-Linear Regression using an Ordinary Least Squares model that attempts to predict annual salary using a combination of the provided features.

Data Preprocessing

Renaming Long Column Names:

```
import pandas

dataRaw = pandas.read_excel('salaryData.xlsx')
dataNamed = dataRaw.rename(columns = {
    'Annual Salary (thousands of dollars)': 'salary',
    'University Type': 'univType',
    'Undergraduate Major': 'major',
    'Gender': 'gender',
    'Years Since Graduation from College': 'yearsOut',
    'Cost of Undergraduate Eduction (thousands of dollars)': 'cost'})
dataNamed.major = dataNamed.major.astype('category').cat.rename_categories({'STEM': 'stem', 'non-STEM': 'nonStem'})
```

Data Table:

```
from tabulate import tabulate

def orgTable(data):
    print(tabulate(data, headers='keys', tablefmt='orgtbl', showindex=False))

orgTable(dataNamed.head())
```

salary	univType	major	gender	yearsOut	cost
57.5145	Private	stem	Male	13	189.791
59.7486	Private	nonStem	Male	7	217.773
61.3799	Private	stem	Male	13	178.894
45.4764	Private	nonStem	Male	8	212.273
43.5117	Private	stem	Male	9	201.306

Dummy Variables:

In order to use categorical data in our regression model, we need to convert the categories to dummy variables. A dummy will be created for category in each column, however, we only need a dummy for $k-1$ of the categories per column to fully explain the original data. In the code below, we set `drop_first` to `True`, so the first dummy for each column will be dropped.

```
import statsmodels.api as sm

data = pandas.get_dummies(dataNamed, columns=['major', 'gender', 'univType'], drop_first=True)
orgTable(data.head())
```

salary	yearsOut	cost	major_nonStem	gender_Male	univType_Public
57.5145	13	189.791	0	1	0
59.7486	7	217.773	1	1	0
61.3799	13	178.894	0	1	0
45.4764	8	212.273	1	1	0
43.5117	9	201.306	0	1	0

Multi-Linear Regression

Checking for Multicollinearity using Variance Inflation Factor (VIF):

VIF quantifies the severity of multicollinearity in an OLS regression analysis. A VIF of 1 indicates a feature that is not correlated. In general, the larger the VIF, the less reliable the regression results will be.

```
from patsy import dmatrices
from statsmodels.stats.outliers_influence import variance_inflation_factor

y, X = dmatrices(
    'salary ~ yearsOut + major_nonStem + gender_Male + cost + univType_Public',
    data=data,
    return_type="dataframe")

def vifDataFrame(X):
    vif = pandas.DataFrame()
    vif['VIF Factor'] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]
    vif['features'] = X.columns
    return vif

orgTable(vifDataFrame(X).round(1))
```

VIF Factor	features
1352.7	Intercept
7	yearsOut
1	major_nonStem
1.1	gender_Male
91.8	cost
89.8	univType_Public

From the table, `cost` and `univType_Public` have high VIFs. We need to remove one of the collinear features. For now, we remove the type of university.

```
y, X = dmatrices(
    'salary ~ yearsOut + major_nonStem + gender_Male + cost',
    data=data,
    return_type="dataframe")

orgTable(vifDataFrame(X).round(1))
```

VIF Factor	features
17.6	Intercept
1.1	yearsOut
1	major_nonStem
1	gender_Male
1	cost

It seems that the cost of the university and the type of university were highly correlated. Consequently, small changes in the data used for estimation could cause wild swings in estimated coefficients. By removing the type of university as a feature, the VIFs are much closer to 1, and the stability of the model should increase.

Backward Elimination:

Now, we will eliminate features that have a p-value greater than our significance level of 0.05.

```
model = sm.OLS(y, X).fit()

def orgTableFromSeries(series, name):
    columns = {'features': series.index, name: series.values}
    dataFrame = pandas.DataFrame(columns)
    orgTable(dataFrame)

orgTableFromSeries(model.pvalues, 'p-values')
```

features	p-values
Intercept	1.30569e-14
yearsOut	1.70023e-06
major_nonStem	0.0749476
gender_Male	0.790979
cost	8.35814e-11

The resulting p-values show that **gender** does not have a significant effect on the annual salary for the participants. So, we eliminate `gender` from the model features.

Next, we re-run OLS, without the `gender` feature, and show the p-values.

```
y, X = dmatrices(
    'salary ~ yearsOut + major_nonStem + cost',
    data=data,
    return_type="dataframe")

model = sm.OLS(y, X).fit()

orgTableFromSeries(model.pvalues, 'p-values')
```

features	p-values
Intercept	3.54395e-16
yearsOut	1.17076e-06
major_nonStem	0.0727082
cost	5.87392e-11

We find that the **major** (stem or non-stem), also does not have a significant effect on the annual salary of participants. As such, we eliminate `major` from the model features.

Hypothesis for Model with only Significant Features:
We believe that all insignificant features have been eliminated. Currently, our model is:

$$\texttt{salary} = \beta_0 + \beta_1 \cdot \texttt{yearsOut} + \beta_2 \cdot \texttt{cost}.$$

Our hypothesis is

$$H_0 : \rho_i = 0 \text{ versus } H_A : \rho_i \neq 0 \text{ for each feature, } i.$$

Once, again we run OLS, this time without the `major` feature.

```
y, X = dmatrices(
    'salary ~ yearsOut + cost',
    data=data,
    return_type="dataframe")

model = sm.OLS(y, X).fit()

orgTableFromSeries(model.pvalues, 'p-values')
```

features	p-values
Intercept	8.50287e-16
yearsOut	1.66187e-06
cost	8.35224e-11

Results of OLS:
Since the p-value for each feature is below the significance level of 0.05, we reject the null hypothesis, and find the number of years out of college and cost of university to have significant effects on the annual salary of participants. That is, the evidence suggests a significant multi-linear trend in the data using the `cost` and `yearsOut` features.

Concrete Fitted Model:
After the backward elimination of features, we obtained a model containing only significant features. We get the following coefficients and R^2 for our model.

```
orgTableFromSeries(model.params, '\(\\beta_i\)')
print(f'|-\n| \(R^2\) | {model.rsquared:.2f} |')
```

features	β_i
Intercept	25.8469
yearsOut	0.944391
cost	0.0776286
R^2	0.56

The fitted model, which has an R^2 of 56%, is

$$\boxed{\hat{\texttt{salary}} = 25.85 + 0.9444 \cdot \texttt{yearsOut} + 0.07763 \cdot \texttt{cost}.}$$

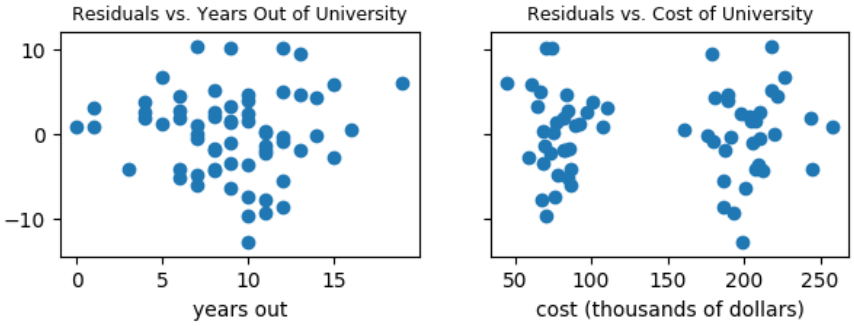
Interpretation of Model:
From our R^2, 56% of the variance in the annual salary is explained by the cost of the university and the number of years out of university the participants are.

If `yearsOut` increases by 1, the predicted salary will increase by $\$944.4$. If the `cost` of university (in thousands of dollars) increases by 1, the predicted salary will increase by $\$77.63$. Furthermore, the intercept is not directly interpretable because the data does not consider participants with an annual salary of $\$0$.

Checking Residuals:

```
from matplotlib import pyplot
figure, axes = pyplot.subplots(1, 2, figsize=(7, 2), sharey=True)
axes[0].scatter(data['yearsOut'], model.resid)
axes[0].set_title('Residuals vs. Years Out of University', fontsize=9)
axes[0].set_xlabel('years out')

axes[1].scatter(data['cost'], model.resid)
axes[1].set_title('Residuals vs. Cost of University', fontsize=9)
axes[1].set_xlabel('cost (thousands of dollars)')
pyplot.savefig('figure/residuals.png', bbox_inches='tight')
```

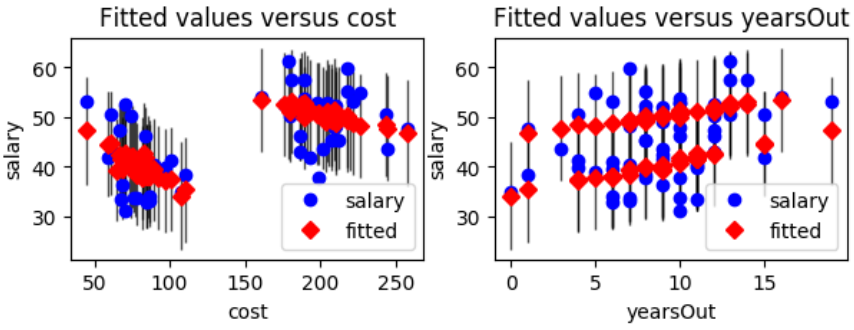


The residuals appear homoscedastic, as the variance is evenly spread and there are no obvious patterns or outliers. This helps verify that a linear model is appropriate for our data.

Model Prediction

Fit Plots:
The following plots illustrate the observed values compared to fitted values.

```
from matplotlib import pyplot
figure, axes = pyplot.subplots(1, 2, figsize=(7, 2))
sm.graphics.plot_fit(model, 'cost', ax=axes[0])
sm.graphics.plot_fit(model, 'yearsOut', ax=axes[1])
pyplot.savefig('figure/fit.png', bbox_inches='tight')
```



Predicting Starting Salary of Males and Females at Clarkson:
Since we eliminated the `gender` feature during backward elimination of features, we cannot use our model to seperately predict the starting salary of males and females. As shown in our analysis, the difference in salary is not significant enough for us to include `gender` as a feature. The predicted salary we will show includes both men and women.

Clarkson's net price, was about $\$31,050$ in 2019. The average starting salary in 2019 was $\$62,756$ –which is 12% above the national average. Now we will use our model to predict a value around the average starting salary in 2019 for Clarkson University undergraduates.

```
def salary(yearsOut, cost):
    return 25.85 + 0.9444*yearsOut + 0.07763*cost

startingSalary = salary(yearsOut = 0, cost = 31.050*4)*1000
print(f'\[\\texttt{{starting salary}} = \${round(startingSalary)}\]')
```

$$\texttt{starting salary} = \$35492$$

The fact that the predicted starting salary is almost half the average starting salary in 2019, is not surprising. Only one participant we used in our OLS had a salary above $60,000 and that participant had 13 years out of university. Likely, our model is missing features that could explain more of the variance in annual salary. It is also possible that we simply do not have enough data, or the data we are using does not fully represent the population (in fact, we do not have any information as to where the data came from).

sources:
https://www.clarkson.edu/news/high-placement-rate-above-average-salaries-clarkson-university-class-2019
https://www.collegesimply.com/colleges/new-york/clarkson-university/price/

Model Summary

```
print(model.summary())
```

OLS Regression Results

Dep. Variable:	salary	R-squared:	0.560
Model:	OLS	Adj. R-squared:	0.546
Method:	Least Squares	F-statistic:	38.83
Date:	Sun, 26 Apr 2020	Prob (F-statistic):	1.33e-11
Time:	23:05:21	Log-Likelihood:	-193.24
No. Observations:	64	AIC:	392.5
Df Residuals:	61	BIC:	399.0
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	25.8469	2.392	10.805	0.000	21.063	30.630
yearsOut	0.9444	0.178	5.304	0.000	0.588	1.300
cost	0.0776	0.010	7.838	0.000	0.058	0.097

Omnibus:	0.286	Durbin-Watson:	2.114
Prob(Omnibus):	0.867	Jarque-Bera (JB):	0.249
Skew:	-0.142	Prob(JB):	0.883
Kurtosis:	2.889	Cond. No.	588.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.