



Capstone - Submission 2

-Chetan R Deshpande



<u>Content</u>	<u>Page No.</u>
1. Model building and Interpretation	2-7
2. Model Tuning	7-13

1. Model building and interpretation -

The first step in our model building process is to import `train_test_split` from `sklearn.model_selection` to split our data in 'Train' and 'Test' buckets.

Next, we assign 'X' with all the features except the target variable and 'y' with the target variable which is the 'Expected CTC.'

Further, we have split the train-test data into 70:30 ratio respectively.

The following is the shape of the data after the split -

X_train - (16825, 28)

X_test - (7212, 28)

y_train - (16825,)

y_test - (7212,)

Now, to build the OLS model we import `statsmodels.api` as `sm` and from `sklearn.metrics` we import `mean_absolute_error`, `mean_squared_error` to evaluate the model.

After adding constants to the train and test data, we fit our `y_train` and `X_train` to our OLS model.

OLS Regression Results

Dep. Variable:	Expected_CTC	R-squared:	0.989
Model:	OLS	Adj. R-squared:	0.989
Method:	Least Squares	F-statistic:	5.797e+04
Date:	Thu, 11 Jan 2024	Prob (F-statistic):	0.00
Time:	14:32:39	Log-Likelihood:	-2.2058e+05
No. Observations:	16825	AIC:	4.412e+05
Df Residuals:	16798	BIC:	4.414e+05
Df Model:	26		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	7.232e+06	7.32e+05	9.882	0.000	5.8e+06	8.67e+06
Total_Experience	-8847.2211	330.142	-26.798	0.000	-9494.335	-8200.107
Total_Experience_in_field_applied	58.0609	205.012	0.283	0.777	-343.785	459.907
Passing_Year_Of_Graduation	-2275.3813	347.564	-6.547	0.000	-2956.643	-1594.120
Passing_Year_Of_PG	1179.3764	210.419	5.605	0.000	766.933	1591.820
Passing_Year_Of_PHD	-2476.1693	213.401	-11.603	0.000	-2894.459	-2057.880
Current_CTC	1.3284	0.002	605.626	0.000	1.324	1.333
No_Of_Companies_worked	478.4094	617.848	0.774	0.439	-732.637	1689.456
Number_of_Publications	-1808.5674	485.684	-3.724	0.000	-2760.559	-856.576
Certifications	-7709.7013	1235.164	-6.242	0.000	-1.01e+04	-5288.649
International_degree_any	4.256e-10	2.9e-11	14.677	0.000	3.69e-10	4.82e-10
Department_Encoded	-517.8273	283.875	-1.824	0.068	-1074.252	38.597
Role_Encoded	25.6227	128.795	0.199	0.842	-226.829	278.074
Industry_Encoded	-11.1964	291.649	-0.038	0.969	-582.859	560.467
Designation_Encoded	-87.4834	190.832	-0.458	0.647	-461.535	286.568
Education_Encoded	-4.611e+04	1272.821	-36.227	0.000	-4.86e+04	-4.36e+04
Graduation_Specialization_Encoded	65.4174	579.583	0.113	0.910	-1070.626	1201.461
University_Grad_Encoded	-542.7658	142.654	-3.805	0.000	-822.383	-263.149
PG_Specialization_Encoded	-285.9817	668.067	-0.428	0.669	-1595.463	1023.499
University_PG_Encoded	-542.7658	142.654	-3.805	0.000	-822.383	-263.149
PHD_Specialization_Encoded	199.9087	553.296	0.361	0.718	-884.611	1284.428

The initial OLS model yielded a high R-squared value of 0.989, indicating strong explanatory power. Additionally, examination of variable p-values provides insights into their individual significance for predicting the target variable in our business analysis.

In our pursuit of accurate salary predictions crucial for decision-making, addressing multicollinearity is imperative. Utilizing `variance_inflation_factor` from `statsmodels.stats.outliers_influence`, we intend to identify and eliminate features with high variance inflation factors (VIF).

VIF values for all columns:

	VIF
const	356925.400746
Total_Experience	6.318427
Passing_Year_Of_Graduation	4.432393
Current_CTC	3.752705
Graduation_Specialization_Encoded	3.301152
PG_Specialization_Encoded	3.238137
Passing_Year_Of_PG	2.671185
Passing_Year_Of_PHD	1.842754
PHD_Specialization_Encoded	1.792865
Total_Experience_in_field_applied	1.637165
Certifications	1.294275
Inhand_Offer_Encoded	1.182177
Last_Appraisal_Rating_Encoded	1.105435
Department_Encoded	1.017538
Role_Encoded	1.007669
Curent_Location_Encoded	1.003139
Designation_Encoded	1.002661
Industry_Encoded	1.001605
Preferred_location_Encoded	1.000981
Organization_Encoded	1.000972

Following the identification and removal of columns exhibiting high variance inflation factors (VIF), these finalized sets of columns will be utilized for subsequent analysis.

OLS Regression Results						
=====						
Dep. Variable:	Expected_CTC	R-squared:	0.988			
Model:	OLS	Adj. R-squared:	0.988			
Method:	Least Squares	F-statistic:	7.294e+04			
Date:	Sat, 13 Jan 2024	Prob (F-statistic):	0.00			
Time:	23:35:00	Log-Likelihood:	-2.2128e+05			
No. Observations:	16825	AIC:	4.426e+05			
Df Residuals:	16805	BIC:	4.428e+05			
Df Model:	19					
Covariance Type:	nonrobust					
=====						
	coef	std err	t	P> t	[0.025	0.975]

const	3.558e+06	5.74e+05	6.194	0.000	2.43e+06	4.68e+06
Total_Experience	-9525.2152	334.574	-28.470	0.000	-1.02e+04	-8869.416
Total_Experience_in_field_applied	184.3297	212.341	0.868	0.385	-231.882	600.541
Passing_Year_Of_Graduation	369.1238	277.809	1.329	0.184	-175.411	913.658
Passing_Year_Of_PG	-120.7922	196.330	-0.615	0.538	-505.620	264.035
Passing_Year_Of_PHD	-2022.4974	188.672	-10.720	0.000	-2392.314	-1652.681
Current_CTC	1.3481	0.002	632.831	0.000	1.344	1.352
Certifications	-2.285e+04	1190.434	-19.197	0.000	-2.52e+04	-2.05e+04
Department_Encoded	-768.6232	295.673	-2.600	0.009	-1348.174	-189.072
Role_Encoded	35.6839	134.174	0.266	0.790	-227.311	298.679
Industry_Encoded	107.9979	303.915	0.355	0.722	-487.708	703.704
Designation_Encoded	7.6195	198.822	0.038	0.969	-382.092	397.331
Graduation_Specialization_Encoded	2628.3392	579.343	4.537	0.000	1492.766	3763.912
PG_Specialization_Encoded	-1893.8125	678.971	-2.789	0.005	-3224.668	-562.957
PHD_Specialization_Encoded	-529.8518	575.220	-0.921	0.357	-1657.344	597.640
Current_Location_Encoded	97.4216	223.512	0.436	0.663	-340.685	535.528
Preferred_location_Encoded	-271.0297	222.929	-1.216	0.224	-707.994	165.935
Inhand_Offer_Encoded	1.442e+05	2252.457	63.998	0.000	1.4e+05	1.49e+05
Last_Appraisal_Rating_Encoded	-2.886e+04	735.800	-39.223	0.000	-3.03e+04	-2.74e+04
Organization_Encoded	-127.4150	208.578	-0.611	0.541	-536.250	281.420

In this revised OLS model, we will assess the significance of variables by examining their p-values. Variables with p-values exceeding 0.05 will be eliminated, focusing on retaining statistically significant predictors.

OLS Regression Results						
=====						
Dep. Variable:	Expected_CTC	R-squared:	0.988			
Model:	OLS	Adj. R-squared:	0.988			
Method:	Least Squares	F-statistic:	1.732e+05			
Date:	Sat, 13 Jan 2024	Prob (F-statistic):	0.00			
Time:	23:35:00	Log-Likelihood:	-2.2129e+05			
No. Observations:	16825	AIC:	4.426e+05			
Df Residuals:	16816	BIC:	4.427e+05			
Df Model:	8					
Covariance Type:	nonrobust					
=====						
	coef	std err	t	P> t	[0.025	0.975]
const	3.927e+06	3.63e+05	10.814	0.000	3.22e+06	4.64e+06
Total_Experience	-9594.2251	270.158	-35.513	0.000	-1.01e+04	-9064.686
Passing_Year_Of_PHD	-1961.2439	180.678	-10.855	0.000	-2315.392	-1607.095
Current_CTC	1.3478	0.002	641.329	0.000	1.344	1.352
Certifications	-2.33e+04	1152.588	-20.214	0.000	-2.56e+04	-2.1e+04
Graduation_Specialization_Encoded	2863.9642	532.307	5.380	0.000	1820.587	3907.342
PG_Specialization_Encoded	-2400.9976	603.468	-3.979	0.000	-3583.858	-1218.138
Inhand_Offer_Encoded	1.44e+05	2211.512	65.111	0.000	1.4e+05	1.48e+05
Last_Appraisal_Rating_Encoded	-2.886e+04	734.500	-39.291	0.000	-3.03e+04	-2.74e+04
=====						
Omnibus:	42.386	Durbin-Watson:	2.014			
Prob(Omnibus):	0.000	Jarque-Bera (JB):	51.496			
Skew:	0.047	Prob(JB):	6.57e-12			
Kurtosis:	3.254	Cond. No.	7.64e+08			

The final columns selected for Expected CTC model building, based on their significance (p-value < 0.05), are:

- **Total_Experience**
- **Passing_Year_Of_PHD**
- **Current_CTC**
- **Certifications**
- **Graduation_Specialization_Encoded**
- **PG_Specialization_Encoded**
- **Inhand_Offer_Encoded**
- **Last_Appraisal_Rating_Encoded**

Now, We will explore a range of models to identify the most effective one -

a. Linear Regression -

Initially, we import the LinearRegression module from sklearn.linear_model and train the model with our designated training data. Following the prediction phase, we assess the model's performance using R-squared for accuracy measurement and RMSE for error evaluation.

The outcomes from our Linear Regression predictions are as follows:

- **Training Set R-squared: 0.98801**
- **Test Set R-squared: 0.98731**
- **Root Mean Squared Error on training data - 124680.64227**
- **Root Mean Squared Error on testing data - 126066.47115**

The Linear Regression model exhibits robust performance, as evidenced by high R-squared values of 98.8% for the training set and 98.7% for the testing set. Additionally, low Root Mean Squared Error values of around 124681 for training and 126066 for testing indicate accurate predictions and minimal errors. This underscores the model's effectiveness in capturing data patterns and making reliable predictions with high precision.

b. Decision Tree -

In initiating the model building process, we import DecisionTreeRegressor from sklearn.tree. Subsequently, the model is trained using our designated training data, and predictions are made accordingly. The model's performance is evaluated through the following metrics:

- **Training Set R-squared: 1.0**
- **Test Set R-squared: 0.99417**
- **Root Mean Squared Error on Training Data: 0.0**
- **Root Mean Squared Error on Testing Data: 85421.25047**

The Decision Tree Regressor model achieves outstanding performance, exhibiting a perfect fit (R-squared of 1.0) to the training data and exceptional predictive accuracy (99.4% R-squared) on the testing data. The model's ability to achieve an RMSE of 0.0 on the training set indicates minimal errors in predictions, while a testing set RMSE of approximately 85421 underscores its accuracy in capturing underlying patterns and making precise predictions on new data. The model demonstrates robust generalization and a high level of efficacy in predicting the Expected_CTC.

c. Random Forest

To initiate the model building process, we import RandomForestRegressor from sklearn.ensemble. Subsequently, the model is trained using the designated training data, and predictions are made on both the training and testing datasets. The evaluation results are as follows:

- **R-squared on Training Data: 0.99958**
- **R-squared on Testing Data: 0.99680**
- **Root Mean Squared Error on Training Data: 23147.59528**
- **Root Mean Squared Error on Testing Data: 63253.78962**

The Random Forest Regressor demonstrates exceptional performance in predicting Expected_CTC, with a high R-squared of 99.96% on the training data and 99.68% on the testing data. The model's accuracy is further underscored by low Root Mean Squared Error values, measuring at 23147.60 for the training set and 63253.79 for the testing set. These results indicate the model's robust ability to capture underlying patterns in the data and make precise predictions, making it a strong candidate for accurate Expected_CTC estimation in our business context.

d. KNN Regressor -

Commencing the model construction, we import KNeighborsRegressor from sklearn.neighbors. Optimal k neighbors, determined through preliminary data processing, are set to 3. Subsequently, the model is trained with the designated training data, and predictions are made for both the training and testing datasets. The evaluation results are presented as follows:

- **Training Set R-squared: 0.99001**
- **Testing Set R-squared: 0.97865**
- **Training Set RMSE: 113758.78681**
- **Testing Set RMSE: 163573.41361**

The K-Nearest Neighbors (KNN) Regressor model, with an optimal k of 3, demonstrates commendable performance. The R-squared values stand at 99.0% for the training set and 97.9% for the testing set, indicating strong predictive accuracy. The Root Mean Squared Error values, measuring at 113758.79 for the training set and 163573.41 for the testing set, suggest a relatively low level of prediction errors. This underscores the model's ability to effectively capture data patterns and make accurate predictions, making it a valuable tool for Expected_CTC estimation in our business context.

2. Model Tuning -

We will construct ensemble models using AdaBoost and XGBoost, leveraging their individual strengths to enhance predictive accuracy. These ensemble approaches aim to capitalize on the diverse learning strategies embedded in AdaBoost and XGBoost, collectively harnessing their capabilities for an even more robust and accurate Expected_CTC prediction.

e. Boosting - AdaBoost -

To initiate the model building process, AdaBoostRegressor from sklearn.ensemble is imported. Subsequently, the model is fitted with the training data, and predictions are made for both the training and testing datasets. The evaluation results, reflecting accuracy and errors in the prediction, are presented below:

- **R-squared on Training Data: 0.98206**
- **R-squared on Testing Data: 0.98117**
- **Root Mean Squared Error on Training Data: 152473.8623**
- **Root Mean Squared Error on Testing Data: 153620.9295**

The AdaBoost Regressor model demonstrates strong predictive performance with R-squared values of 98.2% on the training set and 98.1% on the testing set. Additionally, the Root Mean Squared Error values stand at 152473.86 for the training set and 153620.93 for the testing set, indicating a relatively low level of prediction errors. These results underscore the model's effectiveness in capturing data patterns and making accurate predictions, establishing it as a valuable tool for Expected_CTC estimation in our business context.

f. Gradient Boosting - XGBoost -

To commence, the XGBoost module is imported, followed by the fitting of training data into the model for prediction. Subsequently, the model's evaluation is performed to assess accuracy and errors in the predictions:

- **R-squared on Training Data: 0.9984**
- **R-squared on Testing Data: 0.9971**
- **Root Mean Squared Error on Training Data: 44178.0690**
- **Root Mean Squared Error on Testing Data: 60107.49641**

The XGBoost model exhibits outstanding performance in predicting Expected_CTC. Notably, it achieves high R-squared values of 99.84% on the training set and 99.71% on the testing set, indicating exceptional accuracy. The Root Mean Squared Error values further emphasize the model's precision, measuring at 44178.07 for the training set and 60107.50 for the testing set. These results highlight the efficacy of the XGBoost model in capturing underlying patterns and making accurate predictions, establishing it as a robust tool for Expected_CTC estimation in our business context.

Interpretation -

The following table summarizes the key evaluation metrics for each model, including R-squared values for both training and testing datasets and Root Mean Squared Error values. The results offer insights into the accuracy and predictive performance of each model, aiding in the selection of the most suitable approach for Expected_CTC estimation.

	R-squared Training Data	R-squared Test Data	RMSE on Training date	RMSE on Testing date
Linear Regression	0.988010	0.987320	124680.642278	126066.471156
Decision Tree	1.000000	0.994178	0.000000	85421.250473
Random Forest	0.999587	0.996808	23147.595289	63253.789629
KNN	0.990019	0.978652	113758.786812	163573.413611
ADA Boost	0.982069	0.981171	152473.862376	153620.929592
XG Boost	0.998495	0.997117	44178.069062	60107.496418

In evaluating various regression models for predicting Expected_CTC, several key insights emerge. The Decision Tree model stands out with a perfect R-squared on the training set (1.0) and an impressive 99.4% on the testing set, suggesting an exact fit to the data and excellent generalization capability. Random Forest closely follows, demonstrating a high R-squared of 99.96% on the training set and 99.68% on the testing set, indicating robust predictive accuracy and generalization. XG Boost also performs exceptionally well, achieving R-squared values of 99.84% on the training set and 99.71% on the testing set, highlighting its strong predictive capabilities. Linear Regression, though slightly trailing behind in R-squared, still provides a solid performance with 98.8% on the training set and 98.7% on the testing set. K-Nearest Neighbors (KNN) and ADA Boost, while delivering satisfactory results, exhibit slightly lower predictive accuracy compared to the aforementioned models.

Therefore, based on comprehensive evaluation, we have identified Decision Tree, Random Forest, and XG Boost as the top three models for further exploration in determining the optimal model for predicting Expected_CTC. These models have exhibited outstanding performance in terms of accuracy and predictive capabilities, laying a solid foundation for more in-depth analysis and refinement.

Next, we will **fine-tune our top three models - Random Forest, Decision Tree, and XG Boost** - to further enhance their accuracy in predicting Expected_CTC, ensuring unbiased and effective performance across all scenarios.

a. Decision Tree - Cross-validation tuning -

For Decision Tree cross-validation tuning, we employ RandomizedSearchCV from sklearn.model_selection, fine-tuning the model's parameters systematically. Subsequently, we fit the data into the model, justifying the chosen parameters. Following this, we select the best hyperparameter configuration and predict values on both training and testing data. The resulting evaluation values offer insights into the model's improved performance and accuracy:

- **R-squared for Decision Tree on Training Set: 0.9987**
- **R-squared for Decision Tree on Testing Set: 0.9952**
- **Root Mean Squared Error on Training Set for Decision Tree: 40762.12105**
- **Root Mean Squared Error on Testing Set for Decision Tree: 40762.12105**

The Decision Tree model, after cross-validation tuning, exhibits exceptional performance. The R-squared values indicate a high level of explained variance, with 99.87% on the training set and 99.52% on the testing set. Moreover, the Root Mean Squared Error (RMSE) values are same, measuring at 40762.12 for both the training and testing sets, signifying precise and accurate predictions.

b. Random Forest - Cross-validation - Tuning -

In a similar way as the previous approach, we fit the data into the model, seeking the best hyperparameters through systematic tuning. Subsequently, we predict values on both the training and testing datasets. The ensuing evaluation values provide insights into the model's performance and accuracy:

- **R-squared for Random Forest on Training Set: 0.9992**
- **R-squared for Random Forest on Test Set: 0.9968**
- **Root Mean Squared Error on Training Set for Random Forest: 31709.0100**
- **Root Mean Squared Error on Test Set for Random Forest: 62845.40379**

After fine-tuning the Random Forest model, we observe outstanding performance. The R-squared values are remarkably high, indicating substantial explained variance with 99.92% on the training set and 99.68% on the testing set. Additionally, the Root Mean Squared Error values are notably low, measuring at 31709.01 for the training set and 62845.40 for the testing set. These results signify the model's accuracy and precision in making predictions.

c. Gradient Boosting - XGBoost - Cross-validation - Tuning -

Similarly, for XGBoost, we fit the data into the model, identify the best hyperparameters through systematic tuning, and proceed with predictions. The subsequent evaluation values offer insights into the model's performance and accuracy based on the predicted values:

- R-squared for XGBoost on Training Set: 0.9978
- R-squared for XGBoost on Test Set: 0.9971
- Root Mean Squared Error on Training Set for XGBoost: 52586.64340
- Root Mean Squared Error on Test Set for XGBoost: 59444.01946

The XGBoost model, after fine-tuning, demonstrates commendable performance. The R-squared values are notably high, reflecting substantial explained variance with 99.78% on the training set and 99.71% on the testing set. Additionally, the Root Mean Squared Error values measured at 52586.64 for the training set and 59444.02 for the testing set. These results underscore the model's accuracy and precision in making predictions.

Interpretation -

	R-squared Training Data	R-squared Test Data	RMSE on Training date	RMSE on Testing date
Decision Tree Tuned	0.998718	0.995233	40762.121052	77298.042891
Random Forest Tuned	0.999224	0.996849	31709.010042	62845.403792
XGBoost Tuned	0.997867	0.997181	52586.643406	59444.019469

The above table shows the final tuned models' R-Squared and RMSE values. From these mathematical values, we will be choosing the best model to predict the Expected CTC. These results underscore the effectiveness of the tuning process in enhancing the models' accuracy and predictive capabilities.


The most optimal model for predicting Expected CTC is identified as **XGBoost Tuned**. This choice is substantiated by two key factors:

- i) **High Accuracy:** The model exhibits a remarkable R-squared value, signifying its accuracy in explaining the variance in the data.
- ii) **Well-fit Model:** The minimal difference in RMSE values between the training and testing sets indicates that the model generalizes well and is less prone to overfitting.

Comparing the three models, **XGBoost Tuned** emerges as the preferred choice for predicting Expected CTC, leveraging significant features such as **Total Experience, Passing Year of PHD, Current CTC, Certifications, Graduation Specialization, PG Specialization, Inhand Offer, and Last Appraisal Rating.**

Here are my **top 5 recommendations** on how the XGBoost Tuned model can be helpful to HR in predicting Expected CTC:

- **Strategic Workforce Planning:** HR can utilize the XGBoost Tuned model to forecast Expected CTC for potential hires. This aids in strategic workforce planning by providing HR with insights into the financial implications of hiring decisions, enabling better budgeting and resource allocation.
- **Candidate Offer Negotiation:** HR can employ the model to estimate Expected CTC during the recruitment process. This assists HR in crafting competitive and realistic offers, improving the likelihood of successful negotiations and attracting top talent.
- **Resource Allocation Optimization:** HR can leverage the model's predictions to optimize resource allocation within the organization. HR can use this information to allocate compensation budgets more effectively, ensuring that the available funds are distributed in alignment with talent acquisition goals.
- **Identifying Compensation Discrepancies:** HR can use the model to identify potential discrepancies in expected and actual CTC. By comparing predicted Expected CTC with the actual CTC of current employees, HR can identify individuals whose compensation may deviate significantly from the predicted values, prompting a review and potential adjustment.

- 
- **Performance Benchmarking:** HR can implement the model as a benchmarking tool to assess the competitiveness of the organization's compensation structure. By comparing predicted Expected CTC with industry standards, HR can identify areas where adjustments may be needed to remain competitive in the job market.

Incorporating the **XGBoost Tuned model** into HR processes enhances decision-making by providing data-driven insights into compensation **planning, recruitment, and overall talent management.**