Informe Cornershop: Estimación de tiempo de entrega

Francisca Garay

Como primer approach al problema, decidí empezar por el problema más simple que se me pudo ocurrir. Ocupar sólo una variable para estimar el tiempo. Para esto construí una nueva variable que se llama total_distance la cual representa la distancia (línea recta) entre el local dónde se hace el pedido y el domicilio de entrega (por supuesto esto no es cierto en la realidad ya que uno se mueve en cuadras y no en línea recta). Probé entrenar tres algoritmos de machine learning: Linear Regression, Decision Tree y Random Forest. Lamentablemente, la variable distancia_total tiene una distribución casi plana con respecto a total_minutes (ver plot LR_1D.png) por lo tanto el entrenamiento no tiene muy buenos resultados:

Linear Regression: MAE = 1.73, variance score = 0 (ver plot LR_1D.png)
Decision Tree: MAE=1.74, variance score = -0.01 (ver plot DT_1D.png)
Random Forest Regressor: MAE=1.79, variance score = 0 (ver plot RF_1D.png)

De los tres algoritmos, se puede ver que el mejor Mean Absolute Error (MAE) es de regresión lineal con 1.74 minutos, pero el variance score es cero, lo cuál nos dice que no hay dependencia de esta variable con respecto a total_minutes. Las predicciones temporales sobre la muestra de prueba y la muestra que hay que estimar se pueden ver en los plots (leer leyenda). Tanto para Decision Tree como Random Forest varíamos la profundidad y el número de estimadores, pero lo único que provocó fue empeorar el variance score.

Ahora, veamos qué pasa si agregamos más variables. En total, el dataset a entrenar tiene 16 variables (incuyendo el target total_minutes). Dentro de las variables que cambié esta total_distance (descrita anteriormente), predicted_time y actual_time que las cambié a minutos, on_demand que la cambié a ceros y unos y variables que incluí con la información del dataset de los shoppers (separé la variable por picker y driver): seniority_picker, seniority_driver, found_rate_picker, found_rate_driver, picking_speed_picker, picking_speed_driver, accepted_rate_picker, accepted_rate_driver, rating_picker y rating_driver. No alcancé a agregar información del dataset order_product.

Primero, obtuve el scatter plot de todas las variables (ver scatter_matrix.png) el cuál nos muestra cómo distribuye cada variable con respecto a todas ellas. La más importante a mirar es como se comportan las variables con respecto al target total_minutes. Se puede ver que ninguna de las variables depende mucho de total_minutes, dándonos un mal pronóstico para el entrenamiento. También obtuve la matriz de correlación (ver corr_matrix.png) la cuál nos dice que ninguna de nuestras variables está muy correlacionada con total_minutes. De nuevo un mal pronóstico.

Volví a entrenar los mismos tres algoritmos mencionados anteriormente. Se obtuvo:

Linear Regression: MAE = 2.64, variance score = 0.06
Decision Tree: MAE=2.41, variance score = 0.08
Random Forest Regressor: MAE=2.53, variance score = 0.08

De los tres algoritmos, se puede ver que el mejor Mean Absolute Error (MAE) es de Decision Tree con 2.41 minutos y el variance score sube, con respecto al caso unidimensional, a 0.08, lo cuál nos dice que al incorporar más variables mejora un poco la predicción. Decidí aumentar, de nuevo, la profundidad tanto para Decision Tree como para Random Forest obteniendo:

Decision Tree: MAE=1.93, variance score = -0.35
Random Forest Regressor: MAE=2.53, variance score = 0.29

Se puede ver que Decision Tree bajo su MAE pero su variance score empeoró mucho. Para Random Forest esto ayudó mucho. Su MAE se mantuvo igual pero su variance score mejoró bastante. Sin embargo, todavía está lejos del deseado 1. Aumentando aún más la profundidad y el número de estimadores no mejora mucho más este algoritmo.

Existen muchas maneras de mejorar esto. Se podría probar otros algoritmos como Ridge Regression que usa un término de penalización a los coeficientes o RANSAC que minimiza el efecto de outliers.
También se podría explorar el armar nuevas variables que tengan mejor dependecia con el target y también agregar la información que provee el dataset de order_product (depende mucho qué tipo de orden se haga con el tiempo que toma en llevarla a cabo).