Useful Coding Pipelines (blank w/ annotation)

- Linear Regression

```
lm(response ~ predictor +/*
    predictor, data = ...)
#can specify 0 as reference
    level to make estimates the
    mean instead of difference
    from reference level
```

- Logistic Regression

```
glm(response ~ predictor, data =
     ..., family = binomial)
```

- Poisson Regression

```
glm(response ~ predictor, data =
     ..., family = "poisson")
```

- Visualizations
  - Scatterplot with Estimated Regression

```
... <-
    data %>%
    ggplot(aes(x = ..., y =
        ...)) +
    geom_point() +
    geom_smooth(method = lm/"
        glm", se = FALSE, size
        = 1.5)
```

  - Adding upper and lower limits to histogram (0.05 significance in this example)

```
    ... +
    geom_vline(xintercept =
        quantile(
        lm_boot$boot_slope,
        0.025), col = 'blue',
        size = 1) +
geom_vline(xintercept =
    quantile(lm_boot$boot_slope
    , 0.975), col = 'blue',
    size = 1)
```

- Stuff
  - Tidy w/ confidence interval

```
... <- tidy(lm_here, conf.int
    = FALSE/TRUE, conf.level =
    ...)
#tidy model to get p-value,
    confidence intervals.
    Default is 0.95 and conf.
    int = FALSE
```

  - Tidy to get odds

```
tidy(
    logistic_model_here
    , exponentiate =
    TRUE)
```

  - Bootstrapping w/ Infer

```
bootstrap_slope_infer
    <-
#   ... %>%
#   specify(formula = ... ~ ...)
    %>%
#   generate(reps = ..., type = "
    bootstrap") %>%
#   calculate(stat = "slope")
```

  - Regression Estimates (Fitting Regression to Bootstrap Samples)

```
lm_boot250 <- replicate(sets,
    {
  sample_n(data, size = ...,
     replace = TRUE) %>%
    lm(formula = ... ~ ..., data =
        .) %>%
    .$coef
})
```

```
bootstrap_slope_infer <-
US_cancer_sample250 %>%
specify(formula =
    TARGET_deathRate ~
    povertyPercent) %>%
generate(reps = 1000, type = "
    bootstrap") %>%
calculate(stat = "slope")
```

  - Prediction (Log R using Predict)

```
predict(modelname,
    tibble(predictor1 =
     ..., predictor2 =
    ..., cont.), type =
     ...)
# type = "link" for
    odds, "response"
    for classification
```

  - ANOVA (comparing nested models)

```
anova(model 1, model
    2)
anova(model 1, lm(
    response ~ 1, data
    = ...)) # null
    model comparison
```

  - Split to training and testing pipeline

```
fat_sample <-
fat_sample %>%
mutate(id = row_number())

training_fat <-
    fat_sample %>%
    slice_sample(prop = 0.70,
        replace = FALSE)

selection_set_fat <-
    fat_sample %>%
    anti_join(training_fat, by = "
        id")

training_fat <-
    training_fat %>%
    select(-"id")

selection_set_fat <-
    selection_set_fat %>%
    select(-"id")
```

  - Forward/backward selection:

```
regsubsets(
    x = response ~ .,
    nvmax = num_predictors,
    data = ...,
    method = "backward"/
        forward,
)
```

  - CI for Prediction:

```
properties_sample  %>%
select(assess_val, BLDG_METRE)
    %>%
cbind(predict(lm_sample,
    interval="confidence", se.
    fit=TRUE)$fit)
```

  - Prediction Interval:

```
properties_sample  %>%
select(assess_val, BLDG_METRE)
    %>%
cbind(predict(lm_sample,
    interval="prediction"))
```

  - Training and Testing Split:

```
training_fat <-
fat_sample %>%
slice_sample(prop = 0.7,
    replace = FALSE)

selection_set_fat <-
    fat_sample %>%
    anti_join(training_fat, by = "
        id")

training_fat <-
    training_fat %>%
    select(-"id")

selection_set_fat <-
    selection_set_fat %>%
    select(-"id")
```

- Forward/Backward Selection with regsubsets():

```
fat_backward_sel <- regsubsets(
  x=brozek ~ .,
  nvmax=14,
  data=selection_set_fat,
  method="backward/foward",
)

fat_backward_sel

fat_bwd_summary <- summary(
    fat_backward_sel)
```

- CI for prediction:

```
properties_cip_90 <-
    properties_sample %>%
    select(assess_val, BLDG_METRE)
        %>%
    cbind(predict(lm_sample,
        interval="confidence", level
         = 0.90, se.fit=TRUE)$fit)
```

- Prediction Interval:

```
properties_pi <-
    properties_sample  %>%
    select(assess_val, BLDG_METRE)
        %>%
    cbind(predict(lm_sample,
        interval="prediction", level
         = 0.95 default)
```

- Finding R^2 and MSE using metrics:

```
housing_test_metrics <-
    testing_housing %>%
    metrics(truth = SalePrice,
        estimate = pred_full_OLS)

housing_test_metrics
```

- LASSO using glmnet():

```
lasso_cancer_fitted <- # The
    variable to store the fitted
    model
    glmnet(
        x = cancer_train |> select(-
            lpsa),  # First argument
             is the covariates (note
             that we need at least
            two columns)
        y = cancer_train$lpsa,  #
            Second argument is the
            response
        alpha = 1, # This is the
            default and is
            equivalent to LASSO
        nlambda = 100, # the number
            of lambda values to try
            (default is 100)
        #lambda  = ..., #
            alternatively, the
            values of lambda you
            want try
```

```
        standardize = TRUE, #
            Default value is TRUE
    )
```

- Prediction using LASSO:

```
predict(lasso_cancer_fitted, #
    Fitted model
    newx = cancer_train |> select(-
        svi) |> as.matrix(), # A
        matrix (yes, tibble don't
        work) containing the data
        for prediction,
    s = c(0.4, 0.1) # one or more
        values of lambda
   )
```

- Cross Validation using cv.glm():

```
cv_logistic <-
    cv.glm(
        glmfit =
            breast_cancer_logistic_model
            ,
        data = breast_cancer_train,
        K = 10, cost =
            misclassification_rate )
```

- Confusion Matrix:

```
breast_cancer_confusion_matrix <-
    confusionMatrix(
    data = as.factor(
        breast_cancer_pred_class),
    reference = as.factor(
        breast_cancer_train$target),
    positive = '1'
)
```

- Pairwise frequentist hypothesis testing:

```
pairwise_comparisons <- pairwise.
    prop.test(x = successes,
  n = trials,
  p.adjust.method = "bonferroni/
      pocock/OF",
  alternative = "two.sided/greater/
      less",
)
```

- Getting Pocock critical value:

```
design_pocock <- gsDesign(k = 20, #
    number of interim analysis
    planned
    test.type = 1, # for one-sided
        tests
    delta = 0, # default effect size
    alpha = 0.05, #type I error rate
    beta = 0.2, # type II error rate
    sfu = 'Pocock')

crit_pocock <-
    design_pocock$upper$bound
```

- Getting O'Brien Flemming Critical Values:

```
design_of <- gsDesign(k = 10, #
    number of interim analysis
    planned
    test.type = 1, # for one-sided
        tests
    delta = 0, # default effect size
    alpha = 0.05, #type I error rate
    beta = 0.2, # type II error rate
    sfu = 'OF')

crit_of_10 <- design_of$upper$bound
```

Simple Linear Regression and Multiple Linear Regression

- 3 aspects of modelling:
  - Estimation: Estimate the true unknown relation between response and input variables
  - Inference: Use model to infer information about the unknown relation between variables
  - Prediction: Use model to predict the value of the response for new observations
- General formula for SLR: $E[Y|X] = \beta_1 X + \beta_0 + \epsilon$
  - "For an increase of 1 in input, we'd **expect to observe** an increase of $\beta_1 =$ ... of response **with all other variables constant**". DO NOT say effect/caused (correlation, not causation)
  - Error ϵ: All external factors affecting Y other than X. Assume iid and $E[\epsilon|X] = 0$
  - When determining the best line, choose the line that minimizes the sum of the vertical distance of points to the line (minimize distance of output variable, rather than euclidean distance)
  - Slope estimator $\hat{\beta}_1 = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^{n}(X_i - \bar{X})^2}$
- Additive MLR: Contains categorical and numerical variables, yields two lines with the same slope and different intercept
  - Assume that change in the response per unit change of predictor DOES NOT DEPEND on other predictors
  - Default level is selected by alpha order if not specified. Number of dummy variables is n - 1 (1 less than number of categories)
- Interactive (multiplicative) MLR: Contains categorical and numerical variables, includes product between input variables (covariates) and results in two lines with different slopes and intercepts.
- Using the deathRate povertyPercent + state example:
  - We'd get something like $Y_i = \beta_0 + \beta_1 * 1 + \beta_2 *$ povertyPercent $+ \beta_3 * 1 *$ povertyPercent $+ \epsilon_i$
    - → β_0 is the intercept of our reference line (Indiana)
    - → β_1 is the coefficient of the dummy variable, ie. difference between intercepts of both lines (the 1 means it's an interaction term)
    - → β_2 is the slope of the reference line
    - → β_3 is the difference between slopes of both lines
  - In the text, we used Indiana as our reference for mean mortality, therefore stateKansas = -20.286 means the mean mortality is 20.286 less than Indiana
- A (UNIT NAME) increase in X is **associated** with an **expected** change in Y of β_1 (unit name), keeping all other variables constant

Assumption Violations, Logistic Regression, Poisson Regression

- Reverse causality example: "When parents helped with schoolwork, kids usually performed worse". It may be possible kids not performing well in school received regular parental help.
- Simpson's paradox: Sign of the correlation flip when comparing the entire population vs specific strata
- Confounding variable: Variable that causes change in both the response and at least one input variable
- Establishing causal effects is challenging. It depends on how data were collected and the statistical methods used to analyze the data
- Experimental design: The manner in the randomization of experimental units to treatments is carried out and how data are collected
  - Completely random design (CRD): Experimental units are randomized throughout the data layout - no correlation between observations
  - In CRD, observed and observed confounders are balanced on average
  - Randomized block design (RBD): Splits experimental units into homogeneous blocks to remove variation from nuisance factors, then assigns treatments to each block. Blocks are similar in all aspects except for treatment (ex. subjects of similar age groups are blocks)
  - In RBD, only observed confounders are balanced so only average treatment effects can be estimated
- Observational data: Collecting data by measuring variables/surveying without applying any treatment. Causal effects cannot be established as treatments are not controlled
  - Omitting a confounding factor affects estimates of the regression - including it in the regression solves accuracy issues but may be difficult to do
  - You can stratify to check association within subgroups that share common values of the confounder
- Assumptions of linear regression models (LINEM)
  - L: **Linear** relation
    - → Dubious model if not followed
    - → Can remedy by adding transformed covariates (ex. $X^2, \sqrt{X}.log(X)$)
  - I: Errors are **independent**
    - → Inflates SE (causes errors in CI and hypothesis testing)
    - → Remedy is to adapt model (ex. time series analysis)
  - N: Conditional distribution of the error terms is **Normal** (if errors are iid, thus conditional distribution of response is linear).
    - → Use QQ plot or histogram of residuals
    - → Affects sampling dist, but can use CLT provided n is large or bootstrap
  - E: **Equal variance** of error terms (we assume homoscedasticity, but in reality, a data tend to be heteroscedastic) -¿
    - → Affects SE (and therefore, CI and hypothesis test results)
    - → You can transform response to stabilize variance (ex. $\sqrt{Y}, log(Y)$) but it only works for positive data. Can also bootstrap for CIs and hypothesis tests
  - M: **Multicollinearity**
- Multicollinearity occurs when multiple input variables are correlated - info from one var is masked by another
  - Collinearity is between 2 predictors
  - LS estimator is inflated, very variable, and unreliable
  - SE of LS estimators are large
  - VIF checks increase in SE of coefficients fitted alone vs other variables. If ≥ 5 or 10, suggests multicollinearity
  - You know if removing the variable with the largest VIF worked if the VIFs of other variables decreased
- The bootstrap sampling dist of the slope estimator is tighter as the size of the sample bootstrapped from increases
- Poisson is $log(E[Y|X]) = \beta_0 + \beta_1 * X_1 + ... + \beta_p * X_p$
- Logistic is binary data (ie. y = 1 or 0, $E[Y_i] = P(y_i = 1)$) and predicts the probability of a binary event occurring
  - Logit (log odds): $log(\frac{P(y_i=1|X_i)}{1-P(y_i=1|X_i)}) = \beta_0 + \beta_1 * X_1 + ... + \beta_p * X_p$. As probability increases, odds increases
  - Odds formula: $\frac{P(y_i=1|X_i)}{1-P(y_i=1|X_i)} = e^{\beta_0+\beta_1*X_1+...+\beta_p*X_p}$
  - Alternatively: $P(y_i = 1|X_i) = \frac{e^{\beta_0+\beta_1*X_{i1}}}{1+e^{\beta_0+\beta_1*X_{i1}}}$
  - Here's an example of how to interpret logit:
    - → Intercept: $\beta_0 = -0.9$ If the X was 0 (unit), the logit is expected to be -0.9
    - → Slope: $\beta_1 = 0.01$ An increase of 1 (unit) is associated with an increase in the logit of 0.01
  - Use tidy to find odds or mutate and calculate exponential of estimate exp(estimate)
    - → Intercept: $e^{\beta_0} = 0.4$ if the X was 0 (unit), the odds is expected to be 0.4
    - → Slope: $e^{\beta_1} = 1.1$ it multiplies the odds by 1.1 (ie. odds increases in 10 percent of its value). So if we had sexmale, male passengers odds

were 1.1 times higher than female passengers

- Here's an example of an interactive Logistic Regression:
  - Logit($p_{survival}$) = $\beta_0 + \beta_2$* fare, if $X_{male} = 0$. Logit($p_{survival}$) = $\beta_0 + \beta_1 + (\beta_2 + \beta_3)$* fare, if $X_{male} = 1$
  - Odds $\frac{p_{survival}}{1-p_{survival}} = e^{\beta_0} e^{\beta_2 * fare}$ if $X_{male} = 0$ or $e^{\beta_0+\beta_1} e^{(\beta_2+\beta_3)*fare}$ if $X_{male} = 1$
- Poisson regression models count data, ie. response variable is number of times an event occurs in a fixed time interval (non-negative)
  - $Y_i|X_i$ $Poisson(\lambda_i)$, $E[Y_i|X_i] = \lambda_i$, $Var[Y_i|X_i] = \lambda_i$
  - Models log of expected count, ie. $log(\lambda_i) = \beta_0 + \beta_1 * X_i$
  - An increase in 1 unit X is associated with an expected change in the average number of Y by a factor of $e^{0.0014} = 1.00144$ ie. an increase of 0.14 percent

Model Assessment and Selection, Predictive Modelling

- Absolute measures of fit:
  - RSS = residuals = $\Sigma_{i=1}^{n}(y_i - \hat{y_i})^2$ (sum of difference between finding and prediction squared, ie. difference between actual and prediction squared, aka distance from point to the regression line squared)
  - RSE = residual standard error = $\sqrt{\frac{1}{n-p-1}RSS}$ p is number of covariates
  - Mean squared error = RSS/n
- Relative measures of fit:
  - ESS = explained sum squares = $\Sigma_{i=1}^{n}(\hat{y_i} - \bar{y})^2$. Y-hat predicts y using the LR, y-bar predicts y without a model. If the model is better than nothing, this should be large
  - TSS = total sum squared = $\Sigma_{i=1}^{n}(y_i - \bar{y})^2$ (difference of residuals from the null), if scaled correctly it's the sample variance of Y (ie. difference of each point from the mean squared)
  - TSS = RSS + ESS
- R^2 is ESS/TSS or 1 - RSS/TSS.
  - Proportion of variance (TSS) explained by the model (ESS).
  - It measures in-sample and not test sets (out-of-sample)
  - More input variables = larger regardless of relevance
- Adjusted R^2 = 1 - $\frac{RSS/(n-p-1)}{TSS/(n-1)}$
- Use glance(model) to compute R^2, adjusted R^2, statistic, df, and p value
- glance() equivalencies:

```
anova(lm(response ~ 1, data), lm(
    response ~ explanatory), data))

tidy(lm(response ~ explanatory, data
    )) %>% filter(term == '
    explanatory') %>% pull(p.value)
```

- Nested models are 2 models in which one contains the other model + more coefficients. We test these using the F-statistic (anova)
- Forward selection:
  - Start with intercept only model
  - Select best model of each size and keep adding variables until you get to full model
  - Pick the best one (higher adj r2). We will get an optimal size, pick those with a star in the row of the optimal size
- Backward select is the same, but we start from the full model and remove 1 variable each time
- MSE and RMSE:
  - MSE = mean(prediction - response)^2 (Y units squared)
  - RMSE is the sqrt of MSE (same units as Y)
- BIC penalizes complex models more than AIC. Best model is the one with the lowest AIC/BIC
- CI for prediction: Interval for the true mean response value for given levels of explanatory variables (CI of the line)
  - with 90% confidence, the expected value of a house of size 97 mts is between 301274 and 363407 (rounded)
  - Centred at Y-hat. At lower significance level, it's target (ie. 95% CIP is wider than 90)
  - Narrower when values of covariates are closer to their mean, and wider as covariates move away from the mean
- Prediction interval: Interval the actual response value for a NEW OBSERVATION at the given levels of the covariates (CI of the point).
  - With 95% confidence, the brozek for a man that weighs 154.25lbs will be between 2.824 and 27.072
- Precision: true positive / (true pos + false pos)
- Accuracy: (true pos + true neg) / n
- When interpreting an ANOVA: There is sufficient/insufficient evidence to reject the null hypothesis that model 1 is equivalent to model 2.
- LASSO: Linear regression method iwth regularization (adds penalty term to LoF of the regression)
  - Prevents overfitting and reduces model's variance
  - Penalty is the sum of the absolute values of the coefficients (sets some of them to zero when this value is sufficiently large), it can also deal with multicollinearity
  - May not perform well when some covariates are important but have small coefficients

Predictive Modelling

- LASSO:
  - You must standardize covariates, as LASSO penalizes large coefficients, as such, we must standardize the units of covariates (ie. subtract the mean, then divide result by sd)
  - Fit model using OLS
  - When you increase the penalty λ, the absolute value of each coefficient is reduced. This also improves the RSS
  - When we fit categorical data, we use dummy variables (OHE) this doesn't quite make sense
- LASSO Inference:
  - LASSO gives biased estimates and CIs aren't immediately available
  - You can deal with this using data split. In the first split, run LASSO to select variables. In the second split, fit an OLD and use the inference
  - Splitting loses part of the datset (this increases standard error as sample size decreases)
- You can build a regularized regression using glmnet() and cv.glmnet() (remember, alpha = 0 is ridge, = 1 is LASSO):

```
# step 1: using matrices
Housing_X_train <- as.matrix(
    training_Housing[,-20])
Housing_Y_train <- as.matrix(
    training_Housing[,20])

Housing_X_test <- as.matrix(
    testing_Housing[,-20])
Housing_Y_test <- as.matrix(
    testing_Housing[,20])

# step 2 and 3: fitting LASSO on a
    given grid (try using the
    default as well)
Housing_LASSO <- glmnet(
  x = Housing_X_train, y =
      Housing_Y_train,
  alpha = 1,
  lambda = exp(seq(5, 12, 0.1))
)

#step 4: extracting estimated
    coefficients for a given level
    of regularization
coef(Housing_LASSO, s = 40000 or
    Housing_LASSO$lambda.min)

# step 5: selecting best lambda by
    CV
Housing_cv_LASSO <- cv.glmnet(
  x = Housing_X_train, y =
      Housing_Y_train,
  alpha = 1,
  lambda = exp(seq(5, 12, 0.1))
)

# Full LS predictions
Housing_test_pred_full_OLS <-
    predict(Housing_full_OLS,
    newdata = testing_Housing)

# LASSO predictions with lambda.min
Housing_test_pred_LASSO_min <-
    predict(Housing_cv_LASSO,
    newx = Housing_X_test,
    s = "lambda.min")
```

- Lambda.min provides max AUC, lambda.1se is highest lambda for which the model has an AUC within one SE of the max
- If we split the data and use different parts for model selection and type 1 error, the F-test after the forward selection is closer to the significance level
- The sampling distribution of the lasso estimator is not centered around the true β_1
- LASSO has 2 main problems:
  - Biased estimators: This can be addressed by fitting least squares on the variables selected by LASSO (post-lasso)
  - Post-inference: Fitting an LS regression after LASSO, we use the data to select the variables and to conduct inference. We can't rely on the linear model, unless we split the data
- LASSO can fit an lm on datasets with high multicollinearity (remember, vif ¿= 5 or 10)
- Classification error:
  - Training error is usually less than cv error (unless by some fluke)
- Confusion Matrix provides case counts (success-fail horizontal, success-fail vertical)
  - Sensitivity is number of correct success predictions divided by true successes (true success / true success + false neg) (ideally closer to 1)
  - Specificity is number of correct failure predictions divided by true failures (true neg / true neg + false pos) (ideally closer to 1)
  - Cohen's Kappa: How often predictions and classification coincide by chance. Range from -1 to 1 (ideally, close to 1)
  - If you decrease threshold, specificity decreases and sensitivity increases
- Better ROC arches towards the top left corner (ie. it's better than a random classifier) (1). Bad ROC arches towards the lower right (0).

A/B Testing

- Sequential testing: Data are collected and analyzed in multiple stages, test can be stopped early if a significant difference is observed
  - Increases probability of Type 1 error (multiple tests are being performed)
  - Addressed using group sequential methods.
- Group sequential methods:
  - Bonferroni divides significance level by k (number of tests performed) to have a standard p value across tests. This is really conservative an increases chance of type 2 error. You can also just multiply the p-values you receive, or adjust the critical value

- Pocock sets a symmetric critical value requiring constant significance level for each group.
  - → This will be lower than the Bonferroni cv (less conservative) but higher than the unadjusted (the p value is higher than BF, lower than unadjusted)
  - → As number of peeks increases, crit values increase (type 1 error increases, thus need more conservative CV)
- O'Brien-Fleming is asymmetric boundary with high significance level for the first groups, then gradually decreases for subsequent groups.

- Power analysis estimates the min sample size required given a desired significance, expected difference in means (or just test stat) and power